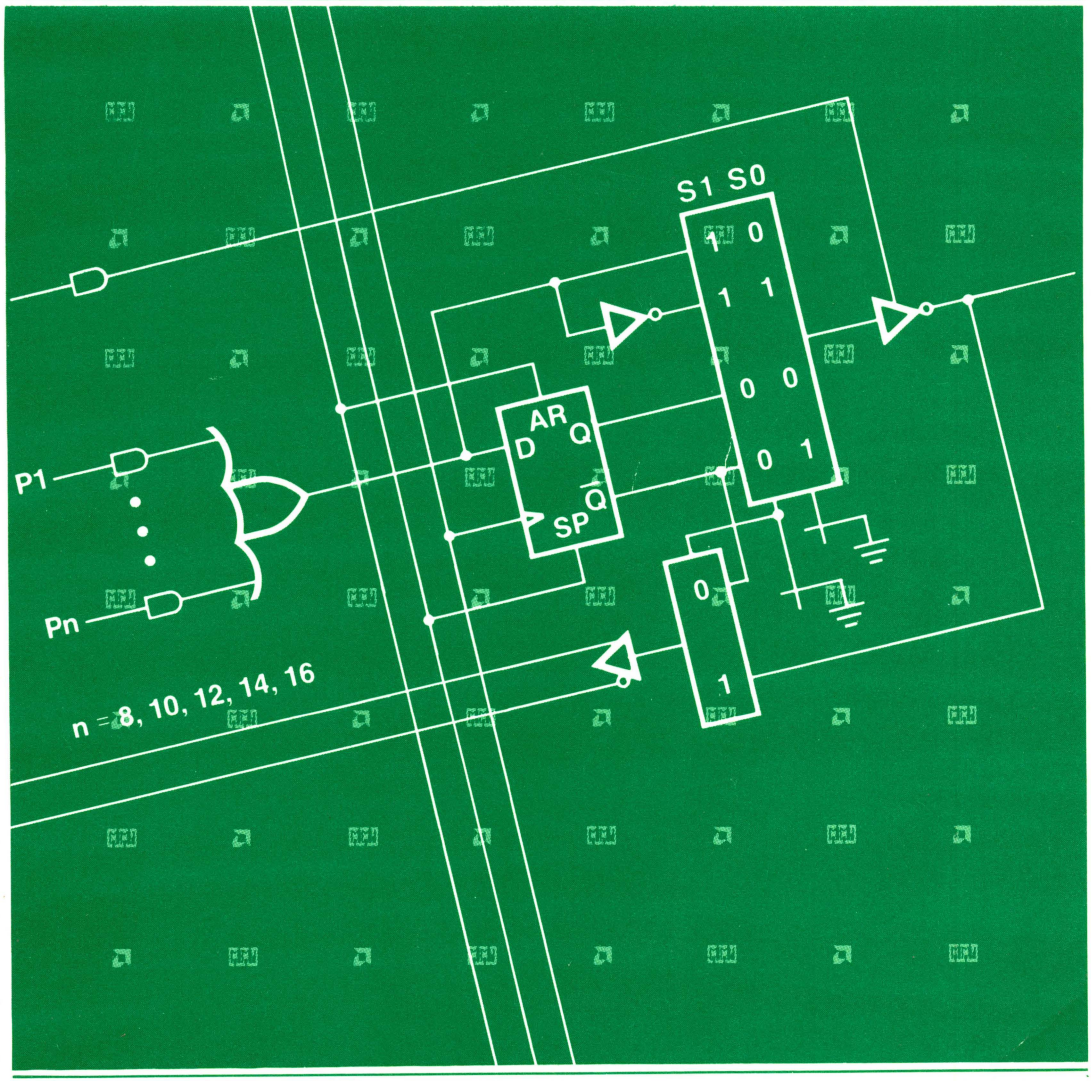
# PAL® Device Data Book

Advanced Micro Devices




Monolithic Memories

A Wholly Owned Subsidiary of Advanced Micro Devices

## 24-pin Combinatorial TTL/CMOS PAL Devices

| PRODUCT | PINS | TECHNOLOGY | $t_{PD}$ (ns) | $I_{CC}$ (mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| AmPAL22XP10 | 24 | TTL | 20, 30, 40 | 90, 180, 210 | XOR gate | 5-271 |
| PAL20S10 | 24 | TTL | 35 | 240 | Product term steering | 5-103 |
| AmPAL22P10 | 24 | TTL | 15, 25 | 105, 210 | 24-pin superset | 5-291 |
| Am/PAL20L10 | 24 | TTL | 15, 20, 25, 30 | 105, 165, 210 | 10 outputs | 5-113, 5-306 |
| PAL20L8 | 24 | TTL, E CMOS | 15, 25, 35, 45 | 0.1, 105, 210 | Standard | 5-122 |
| PAL6L16 | 24 | TTL | 25 | 90 | Wide output | 5-141 |
| PAL8L14 | 24 | TTL | 25 | 90 | Wide output | 5-141 |
| PAL12L10 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |
| PAL14L8 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |
| PAL16L6 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |
| PAL18L4 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |
| PAL20L2 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |
| PAL20C1 | 24 | TTL | 40 | 100 | Simple combinatorial | 5-147 |

## 24-pin Registered TTL/CMOS PAL Devices

| PRODUCT | PINS | TECHNOLOGY | $f_{MAX}$ (MHz) | $I_{CC}$ (mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| AmPALC29MA16 | 24 | EE CMOS | 20, 15 | 120 | Advanced Async. Macro | 5-209 |
| AmPALC29M16 | 24 | EE CMOS | 20, 15 | 120 | Advanced Macrocell | 5-231 |
| PAL32VX10 | 24 | TTL | 25, 22 | 180 | J-K, varied terms | 5-70 |
| Am/PAL22V10 | 24 | TTL, E CMOS | 40, 33, 28.5, 20, 18 | 90, 180 | Versatile | 5-79, 5-249 |
| PAL22RX8 | 24 | TTL | 28.5 | 210 | J-K flip-flops | 5-87 |
| PAL20RA10 | 24 | TTL | 20 | 200 | Asynchronous | 5-95 |
| AmPAL20XRP10 | 24 | TTL | 30, 22, 14 | 105, 180, 210 | XOR gate & polarity | 5-271 |
| AmPAL20XRP8 | 24 | TTL | 30, 22, 14 | 105, 180, 210 | XOR gate & polarity | 5-271 |
| AmPAL20XRP6 | 24 | TTL | 30, 22, 14 | 105, 180, 210 | XOR gate & polarity | 5-271 |
| AmPAL20XRP4 | 24 | TTL | 30, 22, 14 | 105, 180, 210 | XOR gate & polarity | 5-271 |
| PAL20RS10 | 24 | TTL | 20 | 240 | Product term steering | 5-103 |
| PAL20RS8 | 24 | TTL | 20 | 240 | Product term steering | 5-103 |
| PAL20RS4 | 24 | TTL | 20 | 240 | Product term steering | 5-103 |
| PAL20X10 | 24 | TTL | 22 | 180 | XOR gate | 5-113 |
| PAL20X8 | 24 | TTL | 22 | 180 | XOR gate | 5-113 |
| PAL20X4 | 24 | TTL | 22 | 180 | XOR gate | 5-113 |
| AmPAL20RP10 | 24 | TTL | 37, 25 | 105, 210 | Programmable polarity | 5-291 |
| AmPAL20RP8 | 24 | TTL | 37, 25 | 105, 210 | Programmable polarity | 5-291 |
| AmPAL20RP6 | 24 | TTL | 37, 25 | 105, 210 | Programmable polarity | 5-291 |
| AmPAL20RP4 | 24 | TTL | 37, 25 | 105, 210 | Programmable polarity | 5-291 |
| PAL20R8 | 24 | TTL, E CMOS | 37, 25, 20, 15 | 0.1, 105, 210 | Standard | 5-122 |
| PAL20R6 | 24 | TTL, E CMOS | 37, 25, 20, 15 | 0.1, 105, 210 | Standard | 5-122 |
| PAL20R4 | 24 | TTL, E CMOS | 37, 25, 20, 15 | 0.1, 105, 210 | Standard | 5-122 |
| PAL32R16 | 40 | TTL | 16 | 280 | MegaPAL™ device | 5-158 |

(Continued on back cover)

# PAL Device Data Book

## PAL® Device Handbook


| | |
|---|---|
| Introduction | 1 |
| Applications | 2 |


## PAL Device Data Book


| | |
|---|---|
| Programming and Quality | 3 |
| PALASM® 2 Software User Documentation | 4 |
| Data Sheets | 5 |
| Appendices | 6 |

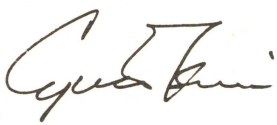

© 1988 Advanced Micro Devices, Inc.

Advanced Micro Devices reserves the right to make changes
in its products without notice in order to improve design or performance characteristics.
The performance characteristics listed in this document are guaranteed by specific tests,
correlated testing, guard banding, design and other practices common to the industry.
For specific testing details contact your local AMD sales representative.
The company assumes no responsibility for the use of any circuits described herein.

901 Thompson Place, P.O. Box 3453, Sunnyvale, California 94088
(408) 732-2400 TWX: 910-339-9280 TELEX: 34-6306

In late 1987, the two programmable logic market leaders, Monolithic Memories and Advanced Micro Devices, merged into one great company. We combined our strong traditions of customer service and innovation to offer you the best line of programmable logic devices. The *1988 Data Book* presents the technical specifications on the entire product line. The separate *1988 Handbook* presents all of the necessary support material, whether you are accustomed to using MMI or AMD products.

AMD/MMI has the products, manufacturing capacity and technology to perpetuate the leadership in the programmable logic field which we pioneered. In the *1988 Data Book* you will notice that we not only have the broadest programmable logic line, but we also have the industry's most comprehensive CMOS programmable logic line. We think that you will find the Data Book and Handbook informative and useful. If you have any comments or questions on the books or the product line, please contact us.



Cyrus Tsui

Vice President
Programmable Logic Division

# Description

This 1988 PAL Device Handbook/Data Book is your complete guide to all programmable logic devices (PLDs) from Monolithic Memories and Advanced Micro Devices. The merger of the two companies provides a greater wealth of products and services for you. Note that all PLDs which were in production before the merger are still being produced.

The PAL Device Handbook/Data Book is organized into two volumes and six easy-to-use sections:

## PAL Device Handbook

### Section 1: Introduction
Includes an overview of the PLD product family.

### Section 2: Applications
Includes detailed application examples. The first few chapters provide tutorials in PLD design. The application notes are grouped by application area.

## PAL Device Data Book

### Section 3: Programming and Quality
Includes information on PLD software programs, programming information, PLD technology and quality discussions, and package information.

### Section 4: PALASM 2 Software User Documentation
Includes complete documentation for PALASM 2 software.

### Section 5: Data Sheets
Includes specifications for all PLDs from the combined company. PAL devices formerly from MMI are under "PAL Devices," while PAL devices formerly from AMD are under "AmPAL Devices."

### Section 6: Appendices
Includes quick reference information.

If you have any questions or comments on PLDs or any other products, please contact your most convenient AMD/MMI sales office, listed at the end of each book.

## Trademarks

PAL®, HAL®, PALASM®, and SKINNYDIP® are registered trademarks of Monolithic Memories, Inc.

ProPAL™, MegaPAL™, ZPAL™, MegaHAL™, ZHAL™, PLE™, PLEASM™, DOC™, Diagnostics-On-Chip™, PROSE™, MONOX™, HiPAC™, and AutoVec™ are trademarks of Monolithic Memories, Inc.

IMOX™ and SSR™ are trademarks of Advanced Micro Devices.

Xilinx™, XACT™, XACTOR™, Logic Cell Array™, and Logic Processor™ are trademarks of Xilinx Inc.

P-SILOS™ is a trademark of SimuCad.

IBM® is a registered trademark of International Business Machines Corporation.

Micro Channel™, IBM-PC™, PC-XT™, and PC-AT™ are trademarks of International Business Machines Corporation.

Data I/O® is a registered trademark of Data I/O Corporation.

ABEL™, PLDtest™, UniSite™, LogicPak™, Logic Fingerprint™, and PROMlink™ are trademarks of Data I/O Corporation.

FutureNet® is a registered trademark of FutureNet, a Data I/O Company.

DASH™, DASH-ABEL™, DASH-GATES™, PLD-CADAT™, and DASH-CADAT-PLUS™ are trademarks of FutureNet, a Data I/O Company.

CUPL™ is a trademark of Personal CAD Systems.

Multibus™ is a trademark of Intel Corporation.

ISDATA® is a registered trademark of ISDATA GmbH.
LOG/iC™ is a trademark of ISDATA GmbH.

Apollo® is a registered trademark of Apollo Computer.

Q-Bus®, UNIBUS®, VAX®, and VMS® are registered trademarks of Digital Equipment Corporation.

Microsoft® is a registered trademark of Microsoft Corporation.
MS-DOS™ is a trademark of Microsoft Corporation.

UNIX™ is a trademark of AT&T Technologies, Inc.

DAISY® and DNIX® are registered trademarks of Daisy Systems.

SOFTPACK™, SOFTLINK™, and LOGILINK™ are trademarks of Digelec, Inc.

ALLPRO™ and LOGIPRO™ are trademarks of Logical Devices, Inc.

PROMAC™ is a trademark of Japan Macnics Corporation.

WordStar™ is a trademark of MicroPro International Corporation.

Mouse System™ is a trademark of Mouse Systems Corporation

TestPLA™ is a trademark of Structured Design, Inc.

NuBus™ is a trademark of Texas Instruments

# Table of Contents

## PAL Device Handbook

### Introduction


**Introduction** ..... 1-1
- What is a PLD? ..... 1-1
- What Advantages do PLDs Have Over Other Implementations? ..... 1-4

**Product Overview** ..... 1-7
- PAL Devices ..... 1-7
- Nomenclature ..... 1-7
- Programmable Sequencers ..... 1-19
- LCA Devices ..... 1-20


### Applications


**Beginner's Guide** ..... 2-1
- Constructing a Combinatorial Design ..... 2-2
- Constructing a Registered Design ..... 2-9
- Programming a Device ..... 2-14

**PLD Design Methodology** ..... 2-21
- Conceptualizing a Design ..... 2-22
- Device Selection Considerations ..... 2-23
- Implementing a Design ..... 2-26
- Simulation ..... 2-30
- Device Programming and Testing ..... 2-33

**Combinatorial Logic Design** ..... 2-35
- Encoders and Decoders ..... 2-35
- Multiplexers ..... 2-43
- Comparators ..... 2-45
- Range Decoders ..... 2-52
- Adders/Arithmetic Circuits ..... 2-53
- Latches ..... 2-58

**Registered Logic Design** ..... 2-61
- Binary Counters ..... 2-67
- Modulo Counters ..... 2-75
- Gray-Code Counters ..... 2-87
- Johnson Counters ..... 2-88
- Shift Registers ..... 2-90
- Asynchronous Registered Designs ..... 2-94

**State Machine Design** ..... 2-101
State Machine Theory ..... 2-103
State Machine Types: Mealy & Moore ..... 2-105
Device Selection Considerations ..... 2-107
PAL Devices as Sequencers ..... 2-111
Programmable Logic Sequencers (PLS) ..... 2-117
PROSE Sequencer (PMS14R21) ..... 2-120
Fuse Programmable Controller (Am29PL141) ..... 2-121
State Machine Design Tutorial ..... 2-122

**Microprocessor-Based Systems** ..... 2-131
Interfacing to the 8086/80186/80286 ..... 2-134
8086 and Am7990 LANCE Interface ..... 2-135
8086 and Am9516 Universal DMA Controller Interface ..... 2-138
80286 to Am9568 Data Ciphering Processor Interface ..... 2-143
80286 to Am8530 Interface ..... 2-147
Interfacing to the 68000/68020 ..... 2-149
The 68000 and Am8530 Interface with Interrupts ..... 2-150
68000 and Am7990 LANCE Interface ..... 2-153
68000 to AmZ8068 Data Ciphering Processor Interface ..... 2-155
68000 and Dual Am9516 DMA Controllers Interface ..... 2-159
Am8530 to 68020 Interface ..... 2-162
Interfacing to the 8088 ..... 2-165
8088 to Am9516 UDC Interface ..... 2-166
80186 to Am9516 Universal DMA Controller Interface ..... 2-170
68000 Interrupt Controller ..... 2-172

**Memory Control** ..... 2-179
Memory Handshake Logic ..... 2-184
Customize a DRAM Controller Using Advanced PAL Devices ..... 2-187
8088 to Am2968 Interface ..... 2-202
MC68000 to Am2968 Interface ..... 2-210
General-Purpose Dual-Port Arbiter ..... 2-215
Dynamic Memory Control State Sequencer ..... 2-224
8-Bit Error Detection and Correction ..... 2-229
Fuse Programmable Controller Simplifies Cache Design ..... 2-239
PAL22RX8A Provides Control and Addressing for a 32-Location-Deep RAM-Based LIFO ..... 2-250

**Graphics and Image Processing Systems** ..... 2-257
Small System Video Controller ..... 2-261
PAL32VX10 Uses Buried Register for Input-Intensive State Machine Designs ..... 2-275

**Digital Signal Processing** ..... 2-283
Waveform Generator ..... 2-286
PAL32VX10 Uses Buried Register for Input-Intensive State Machine Designs ..... 2-292
Analog to Digital Conversion ..... 2-297
PAL Devices, PROMs, FIFOs and Multipliers Team up to
Implement Single-Board High-Performance Audio Spectrum Analyzer ..... 2-307

**Bus Interface** ..... 2-325
Unibus Interrupt Controller ..... 2-328
A MULTIBUS Arbiter Design for 10 MHz Processors ..... 2-333
MULTIBUS to Am9516 Interface ..... 2-338
Z-BUS and 8088/8086 Interface ..... 2-342
VME Bus Control Simplified with PLDs ..... 2-347

# Table of Contents


**Communications** ..... 2-357
- B8ZS Coding Using CMOS ZPAL Devices ..... 2-362
- HDB3 Line Coding Using PAL Devices ..... 2-384
- ZPAL Devices Implement D4 Frame Synchronization ..... 2-404
- T1 Extended Superframe Provides Transmission Error Detection ..... 2-431
- Time Division Multiplexing with the LCA Device ..... 2-435
- LCA Device Implements an 8-Bit Format Converter in a PBX Switching Module ..... 2-444
- Serial Data Link Controller ..... 2-453
- QAM Encoder in a ZPAL Device ..... 2-456
- PAL Devices Implement the Full V.32 Convolution Encoder ..... 2-463
- PLD Devices Implement 4B3T Line Transcoder ..... 2-475
- PALC22V10 Creates Manchester Encoder Circuit ..... 2-499

**Peripheral Design** ..... 2-507
- Building an ESDI Translator Using the M2064 Logic Cell Array ..... 2-509
- Writing a Tape Drive Controller in PROSE ..... 2-519
- GCR (4B-5B) Encoder/Decoder ..... 2-541

**Industrial Control** ..... 2-547
- Stepper Motor Controllers ..... 2-550
- Shaft Encoders ..... 2-558

**Military Applications** ..... 2-579
- Radiation Hardness ..... 2-581

**Article Reprints**
- Programmable Logic Device Preserves Pins, Product Terms (PAL32VX10) ..... 2-589
- PLD Programmability Extends its Sway Over Complex I/O (PALC29M/MA16) ..... 2-593
- PROSE Devices Simplify State Machine Design (PMS14R21) ..... 2-601
- Designing a State Machine with a Programmable Sequencer (PMS14R21) ..... 2-607
- FPCs and PLDs Simplify VME Bus Control (Am29PL141) ..... 2-619
- Fuse-Programmable Chip Takes Command of Distributed Systems (Am29PL141) ..... 2-633
- PAL Device Buries Registers, Brings State Machines to Life (AmPAL23S8) ..... 2-639
- Programmable Event Generator Conquers Timing Restraints (Am2971) ..... 2-644
- Wait-State Remover Improves System Performance (PAL22V10) ..... 2-648
- PLDs Implement Encoder/Decoder for Disk Drives (PAL22V10) ..... 2-651
- Mixing Data Paths Expands Options in System Design (PAL22V10) ..... 2-660
- Programmable Logic Chip Rivals Gate Arrays in Flexibility (PAL22V10) ..... 2-670
- XOR PLDs Simplify Design of Counters and Other Devices (PAL20X10) ..... 2-676
- The PAL20RA10 Story —The Customization of a Standard Product (PAL20RA10) ..... 2-683
- PLDs Abound: RAM-Based Logic Joins In ..... 2-699
- Introduction to Programmable Array Logic ..... 2-702
- Logical Alternatives in Supermini Design ..... 2-711

**Conference Proceedings**
- New PAL Device Architecture Extends Design Flexibility (PAL32VX10) ..... 2-719
- PROSE Architecture and Design Methodology (PMS14R21) ..... 2-724
- Blazing Fast PAL Devices Enable New Application Areas (PAL16R8-10, PAL10H20P/G8) ..... 2-730

**Sales Offices** ..... 2-740

# PAL Device Data Book

## Programming and Quality


**Design Software for Programmable Logic** ..... 3-1
- PALASM 2 Logic Design Software Package ..... 3-5
- PLPL: Programmable Logic Programming Language ..... 3-7
- Logic Cell Array and Development Systems ..... 3-21
- ABEL-GATES ..... 3-35
- CUPL ..... 3-43
- LOG/iC ..... 3-66

**Programming** ..... 3-76
- Programmer Reference Guide ..... 3-81
- ProPAL, HAL and ZHAL Devices Program ..... 3-104

**Testability** ..... 3-108
- Designing Testable Combinatorial Circuits ..... 3-109
- Designing Testable Sequential Circuits ..... 3-114
- Designing Testable State Machines ..... 3-118
- Designing for Testability with the PROSE Device ..... 3-123
- Using Test Vectors ..... 3-128

**Technology and Quality**
- MONOX 3 Oxide-Isolated Process ..... 3-130
- Product Assurance ..... 3-132
- Test and Finish Operations ..... 3-138
- IMOX Product Technology and Reliability ..... 3-140
  - IMOX Product Reliability ..... 3-141
  - IMOX Product Testability ..... 3-144
- ECL Technology ..... 3-150
- CMOS HiPAC Technology ..... 3-155
- CMOS EE Technology ..... 3-159
- Latchup in CMOS Integrated Circuits ..... 3-160
- Metastability ..... 3-164

**Packaging**
- Surface Mount Technology ..... 3-170
- PAL Device Package Outlines ..... 3-179
- PAL Device Package Thermal Characteristics ..... 3-208
- AmPAL Device Package Outlines ..... 3-224
- AmPAL Device Package Thermal Characteristics ..... 3-231


## PALASM 2 Software User Documentation


**Introduction** ..... 4-1
**Install PALASM 2 Software** ..... 4-15
**Run the Software** ..... 4-29
**Build a Boolean Equation Design** ..... 4-61
**Build a State Machine Design** ..... 4-95
**Build Simulation** ..... 4-137
**Program the Device** ..... 4-169
**PALASM 2 Software Glossary** ..... 4-183
**PALASM 2 Software Index** ..... 4-189

## Data Sheets


**PAL/PLD Device Menu** ..... 5-3

**TTL/CMOS PAL Devices** ..... 5-9

PAL16RA8 ..... 5-11

PAL16RP8A Series ..... 5-17
- PAL16P8A
- PAL16RP8A
- PAL16RP6A
- PAL16RP4A

PAL16R8 Family ..... 5-26

PAL16R8D Series ..... 5-29
- PAL16L8D
- PAL16R8D
- PAL16R6D
- PAL16R4D

PAL16R8B Series ..... 5-31
- PAL16L8B
- PAL16R8B
- PAL16R6B
- PAL16R4B

PALC16R8Q-25 Series ..... 5-33
- PAL16L8Q-25
- PAL16R8Q-25
- PAL16R6Q-25
- PAL16R4Q-25

PAL16R8B-2 Series ..... 5-35
- PAL16L8B-2
- PAL16R8B-2
- PAL16R6B-2
- PAL16R4B-2

PAL16R8A Series ..... 5-37
- PAL16L8A
- PAL16R8A
- PAL16R6A
- PAL16R4A

PAL16R8B-4 Series ..... 5-39
- PAL16L8B-4
- PAL16R8B-4
- PAL16R6B-4
- PAL16R4B-4

PAL16R8A-2 Series ..... 5-41
- PAL16L8A-2
- PAL16R8A-2
- PAL16R6A-2
- PAL16R4A-2

PAL16R8A-4 Series ..... 5-43
- PAL16L8A-4
- PAL16R8A-4
- PAL16R6A-4
- PAL16R4A-4

PALC16R8Z-25 Series ..... 5-50
- PAL16L8Z-25
- PAL16R8Z-25
- PAL16R6Z-25
- PAL16R4Z-25

PAL16X4 ..... 5-51

PAL10H8 Series ..... 5-56
- PAL10H8
- PAL12H6
- PAL14H4
- PAL16H2
- PAL16C1
- PAL10L8
- PAL12L6
- PAL14L4
- PAL16L2

PAL32VX10A ..... 5-70

PAL32VX10 ..... 5-70

PALC22V10H-25 ..... 5-79

PALC22V10H-35 ..... 5-79

PAL22RX8A ..... 5-87

PAL20RA10-20 ..... 5-95

PAL20RA10 ..... 5-97

PAL20RS10 Series ..... 5-103
- PAL20S10
- PAL20RS10
- PAL20RS8
- PAL20RS4

PAL20X10A Series ..... 5-113
- PAL20L10A
- PAL20X10A
- PAL20X8A
- PAL20X4A

PAL20R8 Family ..... 5-122
PAL20R8B Series ..... 5-125
PAL20L8B
PAL20R8B
PAL20R6B
PAL20R4B
PAL20R8B-2 Series ..... 5-126
PAL20L8B-2
PAL20R8B-2
PAL20R6B-2
PAL20R4B-2
PAL20R8A Series ..... 5-128
PAL20L8A
PAL20R8A
PAL20R6A
PAL20R4A
PAL20R8A-2 Series ..... 5-130
PAL20L8A-2
PAL20R8A-2
PAL20R6A-2
PAL20R4A-2
PALC20R8Z-35 Series ..... 5-133
PALC20L8Z-35
PALC20R8Z-35
PALC20R6Z-35
PALC20R4Z-35
PALC20R8Z-45 Series ..... 5-133
PALC20L8Z-45
PALC20R8Z-45
PALC20R6Z-45
PALC20R4Z-45

PAL6L16A ..... 5-141
PAL8L14A ..... 5-141

PAL12L10 Series ..... 5-147
PAL12L10
PAL14L8
PAL16L6
PAL18L4
PAL20L2
PAL20C1

PAL32R16 ..... 5-158

General Information ..... 5-164

**TTL/CMOS AmPAL Devices** ..... 5-167

AmPAL23S8-20 ..... 5-169
AmPAL23S8-25 ..... 5-169

AmPAL16R8 Family ..... 5-184
AmPAL16R8D Series ..... 5-183
AmPAL16L8D
AmPAL16R8D
AmPAL16R6D
AmPAL16R4D
AmPAL16R8B Series ..... 5-197
AmPAL16L8B
AmPAL16R8B
AmPAL16R6B
AmPAL16R4B
AmPAL16R8AL Series ..... 5-197
AmPAL16L8AL
AmPAL16R8AL
AmPAL16R6AL
AmPAL16R4AL
AmPAL16R8A Series ..... 5-197
AmPAL16L8A
AmPAL16R8A
AmPAL16R6A
AmPAL16R4A
AmPAL16R8Q Series ..... 5-197
AmPAL16L8Q
AmPAL16R8Q
AmPAL16R6Q
AmPAL16R4Q
AmPAL16R8L Series ..... 5-197
AmPAL16L8L
AmPAL16R8L
AmPAL16R6L
AmPAL16R4L
AmPAL16R8 Series ..... 5-197
AmPAL16L8
AmPAL16R8
AmPAL16R6
AmPAL16R4

AmPAL18P8B ..... 5-202
AmPAL18P8AL ..... 5-202
AmPAL18P8A ..... 5-202
AmPAL18P8Q ..... 5-202
AmPAL18P8L ..... 5-202

AmPALC29MA16-35 ..... 5-209
AmPALC29MA16-45 ..... 5-209
AmPALC29M16-35 ..... 5-231
AmPALC29M16-45 ..... 5-231

AmPAL22V10-15 ..... 5-249
AmPAL22V10A ..... 5-260
AmPAL22V10 ..... 5-260

AmPAL20XRP10 Family ..... 5-271
AmPAL20XRP10-20 Series ..... 5-286
AmPAL22XP10-20
AmPAL20XRP10-20
AmPAL20XRP8-20
AmPAL20XRP6-20
AmPAL20XRP4-20
AmPAL20XRP10-30L Series ..... 5-286
AmPAL22XP10-30L
AmPAL20XRP10-30L
AmPAL20XRP8-30L
AmPAL20XRP6-30L
AmPAL20XRP4-30L
AmPAL20XRP10-30 Series ..... 5-286
AmPAL22XP10-30
AmPAL20XRP10-30
AmPAL20XRP8-30
AmPAL20XRP6-30
AmPAL20XRP4-30
AmPAL20XRP10-40L Series ..... 5-286
AmPAL22XP10-40L
AmPAL20XRP10-40L
AmPAL20XRP8-40L
AmPAL20XRP6-40L
AmPAL20XRP4-40L

AmPAL20RP10 Family ..... 5-291
AmPAL20RP10B Series ..... 5-306
AmPAL22P10B
AmPAL20RP10B
AmPAL20RP8B
AmPAL20RP6B
AmPAL20RP4B
AmPAL20RP10AL Series ..... 5-306
AmPAL22P10AL
AmPAL20RP10AL
AmPAL20RP8AL
AmPAL20RP6AL
AmPAL20RP4AL
AmPAL20RP10A Series ..... 5-306
AmPAL22P10A
AmPAL20RP10A
AmPAL20RP8A
AmPAL20RP6A
AmPAL20RP4A

AmPAL20L10B ..... 5-306
AmPAL20L10-20 ..... 5-306
AmPAL20L10AL ..... 5-306

**PROSE/PLS Sequencers** ..... 5-313
PMS14R21A ..... 5-315
PMS14R21 ..... 5-315

PLS167-33 ..... 5-331
PLS168-33 ..... 5-331
PLS105-37 ..... 5-331

**FPC/PEG Sequencers** ..... 5-337
Am29PL141 Fuse Programmable Controller ..... 5-339
Am2971 Programmable Event Generator ..... 5-365

**ECL PAL Devices** ..... 5-379
PAL10020EV/EG8 ..... 5-381
PAL10H20EV/EG8 ..... 5-381
PAL10H20G8 ..... 5-382
PAL10H20P8 ..... 5-386

**HAL/ZHAL Devices** ..... 5-391
ZHAL20A Series ..... 5-394
ZHAL24A Series ..... 5-401

**Military PAL Devices** ..... 5-415
Introduction ..... 5-417
Military PAL/PLD Device Menu ..... 5-418
Military 20-pin PAL Devices ..... 5-421
Military 24-pin PAL Devices ..... 5-439
DC/AC Parametric Testing ..... 5-469
JAN 38510 and Standard Military Drawings ..... 5-470
Military Screening ..... 5-474
Quality Programs ..... 5-477

**Logic Cell Array** ..... 5-481
M2064 ..... 5-483
M2018 ..... 5-483
Military M2064/M2018 ..... 5-518

**Electrical Definitions** ..... 5-531

## Appendices


Logic Reference ..... 6-1
- Basic Logic Elements ..... 6-1
- Basic Storage Elements ..... 6-8
- Binary Numbers ..... 6-15

Signal Polarity ..... 6-19
Glossary ..... 6-24
Competitive Cross-Reference ..... 6-30
Worldwide Application Support ..... 6-40
Sales Offices ..... 6-41

# Alphanumeric Product Index


Am29PL141 5-339
Am2971 5-365
AmPAL16L8 5-197
AmPAL16L8A 5-197
AmPAL16L8AL 5-197
AmPAL16L8B 5-197
AmPAL16L8D 5-183
AmPAL16L8L 5-197
AmPAL16L8Q 5-197
AmPAL16R4 5-197
AmPAL16R4A 5-197
AmPAL16R4AL 5-197
AmPAL16R4B 5-197
AmPAL16R4D 5-183
AmPAL16R4L 5-197
AmPAL16R4Q 5-197
AmPAL16R6 5-197
AmPAL16R6A 5-197
AmPAL16R6AL 5-197
AmPAL16R6B 5-197
AmPAL16R6D 5-183
AmPAL16R6L 5-197
AmPAL16R6Q 5-197
AmPAL16R8 5-197
AmPAL16R8A 5-197
AmPAL16R8AL 5-197
AmPAL16R8B 5-197
AmPAL16R8D 5-183
AmPAL16R8L 5-197
AmPAL16R8Q 5-197

AmPAL18P8A 5-202
AmPAL18P8AL 5-202
AmPAL18P8B 5-202
AmPAL18P8L 5-202
AmPAL18P8Q 5-202

AmPAL20L10AL 5-306
AmPAL20L10B 5-306
AmPAL20L10-20 5-306
AmPAL20RP4A 5-306
AmPAL20RP4AL 5-306
AmPAL20RP4B 5-306
AmPAL20RP6A 5-306
AmPAL20RP6AL 5-306
AmPAL20RP6B 5-306
AmPAL20RP8A 5-306
AmPAL20RP8AL 5-306
AmPAL20RP8B 5-306
AmPAL20RP10A 5-306
AmPAL20RP10AL 5-306
AmPAL20RP10B 5-306
AmPAL20XRP4-20 5-286
AmPAL20XRP4-30L 5-286
AmPAL20XRP4-30 5-286
AmPAL20XRP4-40L 5-286
AmPAL20XRP6-20 5-286
AmPAL20XRP6-30L 5-286
AmPAL20XRP6-30 5-286
AmPAL20XRP6-40L 5-286
AmPAL20XRP8-20 5-286
AmPAL20XRP8-30L 5-286
AmPAL20XRP8-30 5-286
AmPAL20XRP8-40L 5-286
AmPAL20XRP10-20 5-286
AmPAL20XRP10-30L 5-286
AmPAL20XRP10-30 5-286
AmPAL20XRP10-40L 5-286

AmPAL22P10A 5-306
AmPAL22P10AL 5-306
AmPAL22P10B 5-306
AmPAL22V10 5-260
AmPAL22V10-15 5-249
AmPAL22V10A 5-260
AmPAL22XP10-20 5-286
AmPAL22XP10-30L 5-286
AmPAL22XP10-30 5-286
AmPAL22XP10-40L 5-286

AmPAL23S8-20 5-169
AmPAL23S8-25 5-169

AmPALC29M16-35 5-231
AmPALC29M16-45 5-231
AmPALC29MA16-35 5-209
AmPALC29MA16-45 5-209

M2018 5-483
M2064 5-483

PAL6L16A 5-141
PAL8L14A 5-141

PAL10H8 5-56
PAL10H20G8 5-382
PAL10H20EV/EG8 5-381
PAL10H20P8 5-386
PAL10L8 5-56

PAL12H6 5-56
PAL12L6 5-56
PAL12L10 5-147

PAL14H4 5-56
PAL14L4 5-56
PAL14L8 5-147

PAL16C1 5-56
PAL16H2 5-56
PAL16L2 5-56
PAL16L6 5-147
PAL16L8D 5-29
PAL16L8A 5-37
PAL16L8A-2 5-41
PAL16L8A-4 5-43
PAL16L8B 5-31
PAL16L8B-2 5-35
PAL16L8B-4 5-39
PAL16P8A 5-17
PAL16R4D 5-29
PAL16R4A 5-37
PAL16R4A-2 5-41
PAL16R4A-4 5-43
PAL16R4B 5-31
PAL16R4B-2 5-35
PAL16R4B-4 5-39
PAL16R6D 5-29
PAL16R6A 5-37
PAL16R6A-2 5-41
PAL16R6A-4 5-43
PAL16R6B 5-31
PAL16R6B-2 5-35
PAL16R6B-4 5-39
PAL16R8D 5-29
PAL16R8A 5-37
PAL16R8A-2 5-41
PAL16R8A-4 5-43
PAL16R8B 5-31
PAL16R8B-2 5-35
PAL16R8B-4 5-39
PAL16RA8 5-11
PAL16RP4A 5-17
PAL16RP6A 5-17
PAL16RP8A 5-17
PAL16X4 5-51

PAL18L4 5-147

PAL20C1 5-147
PAL20L2 5-147
PAL20L10A 5-113
PAL20L8A 5-128
PAL20L8A-2 5-130
PAL20L8B 5-125
PAL20L8B-2 5-126
PAL20R4A 5-128
PAL20R4A-2 5-130
PAL20R4B 5-125
PAL20R4B-2 5-126
PAL20R6A 5-128
PAL20R6A-2 5-130
PAL20R6B 5-125
PAL20R6B-2 5-126
PAL20R8A 5-128
PAL20R8A-2 5-130
PAL20R8B 5-125
PAL20R8B-2 5-126
PAL20RA10 5-97
PAL20RA10-20 5-95
PAL20RS10 5-103
PAL20RS4 5-103
PAL20RS8 5-103
PAL20S10 5-103
PAL20X4A 5-113
PAL20X8A 5-113
PAL20X10A 5-113

PAL22RX8A 5-87

PAL32R16 5-158
PAL32VX10 5-70
PAL32VX10A 5-70

PAL10020EV/EG8 5-381

PALC16L8Q-25 5-33
PALC16L8Z-25 5-50
PALC16R4Q-25 5-33
PALC16R4Z-25 5-50
PALC16R6Q-25 5-33
PALC16R6Z-25 5-50
PALC16R8Q-25 5-33
PALC16R8Z-25 5-50
PALC20L8Z-35 5-133
PALC20R4Z-35 5-133
PALC20R6Z-35 5-133
PALC20R8Z-35 5-133
PALC20L8Z-45 5-133
PALC20R4Z-45 5-133
PALC20R6Z-45 5-133
PALC20R8Z-45 5-133
PALC22V10H-25 5-79
PALC22V10H-35 5-79

PLS105-37 5-331
PLS167-33 5-331
PLS168-33 5-331

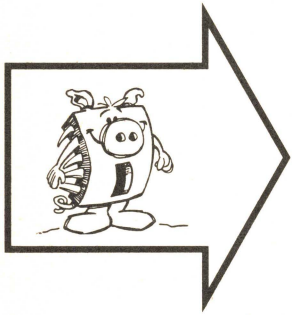
PMS14R21 5-315
PMS14R21A 5-315


IMPORTANT
NOTES

## PAL Device Handbook


Introduction 1

Applications 2

## PAL Device Data Book


Programming and Quality 3

PALASM 2 Software User Documentation 4

Data Sheets 5

Appendices 6

**Design Software for Programmable Logic** ..... 3-1
PALASM 2 Logic Design Software Package ..... 3-5
PLPL: Programmable Logic Programming Language ..... 3-7
Logic Cell Array and Development Systems ..... 3-21
ABEL-GATES ..... 3-35
CUPL ..... 3-43
LOG/iC ..... 3-66

**Programming** ..... 3-76
Programmer Reference Guide ..... 3-81
ProPAL, HAL and ZHAL Devices Program ..... 3-104

**Testability** ..... 3-108
Designing Testable Combinatorial Circuits ..... 3-109
Designing Testable Sequential Circuits ..... 3-114
Designing Testable State Machines ..... 3-118
Designing for Testability with the PROSE Device ..... 3-123
Using Test Vectors ..... 3-128

**Technology and Quality**
MONOX 3 Oxide-Isolated Process ..... 3-130
Product Assurance ..... 3-132
Test and Finish Operations ..... 3-138
IMOX Product Technology and Reliability ..... 3-140
IMOX Product Reliability ..... 3-141
IMOX Product Testability ..... 3-144
ECL Technology ..... 3-150
CMOS HiPAC Technology ..... 3-155
CMOS EE Technology ..... 3-159
Latchup in CMOS Integrated Circuits ..... 3-160
Metastability ..... 3-164

**Packaging**
Surface Mount Technology ..... 3-170
PAL Device Package Outlines ..... 3-179
PAL Device Package Thermal Characteristics ..... 3-208
AmPAL Device Package Outlines ..... 3-224
AmPAL Device Package Thermal Characteristics ..... 3-231

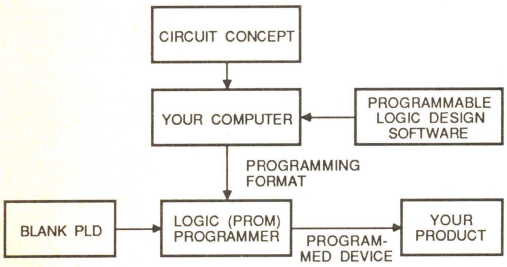
# Design Software for Programmable Logic

## Introduction

Programmable logic design software translates a custom logic design specification into a format which can be accepted by a programmer (Figure 1).

Programmable logic software is also an excellent tool for design simulation and documentation. Simulation assists in debugging an initial design and helps to ensure that a device will operate as intended the first time instead of requiring multiple design iterations. Documentation is essential for someone other than the original designer to understand a custom programmable logic specification.

This overview will describe the basic components of PLD design software packages, including assistance in logic simulation and testing. Several software packages are available. They are listed at the end of this overview, along with references to the appropriate pages for more information or user documentation.




602 01

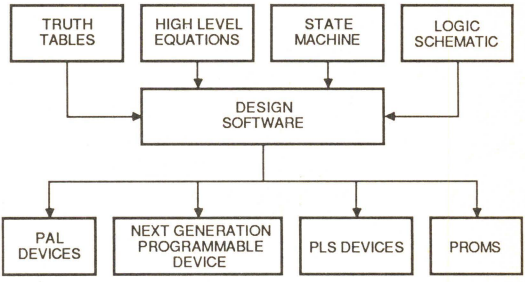
**Figure 1. The Programmable Logic Development Cycle**

## Design Software for Programmable Logic

PLD design software lets the designer write logic descriptions at a high level, that is, at a level that accurately reflects the design concept. This type of software increases productivity while producing designs that are thoroughly documented.

The software should support all programmable logic device types, all popular logic (PROM) programmers, and a large number of popular development computers. In addition, software products offer a variety of input design formats such as state machines, high-level Boolean equations, truth tables and logic schematics.

A compiler's syntax offers a general and easy description of the desired configuration of the chosen programmable logic device (PLD).




602 02

**Figure 2. The Compiler**

In addition, the high-level description of the design provides flexibility in changing the design if so desired. A designer might use a particular type of PLD. Later, when fixes or enhancements are made, the design can be quickly re-compiled for the same device. If the changes require more product terms or an architectural configuration that the chosen PLD cannot support, the function can easily be placed in an alternate device. In many cases this will allow design modifications without altering printed circuit boards which may have already been manufactured.

## Logic Simulation

Most of the PLD software design tools also offer logic simulation. Logic simulation is typically performed to verify the logical design prior to programming an actual device. This may save some of the time spent trouble-shooting a programmable logic design using conventional techniques, using an oscilloscope and logic analyzer.

A simulation file consists of stimulus patterns applied to inputs and response patterns expected at outputs. The simulator compares each stimulus/response pattern, or vector, with the logic equations to verify that the expected response agrees with that produced according to the equations.

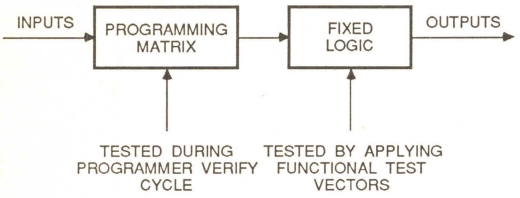
Not simulating may be of little consequence for simple designs, but for complex designs, especially complex sequential logic, it is well worth the time.

## Testing Programmable Logic

PLD software design tools also assist the designer in testing the PLD after it has been programmed.

Before shipping a PLD, programmability may be verified by the manufacturer by exercising the device's address and programming circuitry on redundant test sites.

After the device has been received and programmed by the user, the logic programmer will read the states of all the fuses in the device and compare them with the data stored in the programmer's memory to check the status of the programming matrix, in its verify cycle (Figure 3). If any mismatches are detected, the device is rejected.




602 03

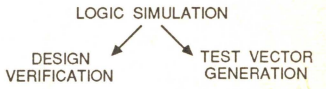
**Figure 3. Programmable Logic Device Testing**

However, a correct fuse verify does not guarantee that the device will work properly, since the fixed logic of the device has not been fully tested. To ensure proper operation the device must be functionally tested.

Functional testing of PLDs involves applying stimulus patterns to a device while looking for the expected response. The test sequence consists of a table of stimulus/response patterns similar to those used to perform a simulation. PLD software design tools offer the capability of generating these test vectors.

Test vectors are produced by creating a simulation input file containing stimulus/response patterns. After running the simulator to verify the integrity of the vectors, they are appended to the JEDEC down-loadable file which already contains the programming patterns for the particular target device.

We can now see that there are two distinct benefits of logic simulation in working with PLDs:




602 04

## Software Tools

Many different programmable logic design aid software programs are available. Table 1 lists some current suppliers of these design tools. Contact the indicated companies for the status of their particular product.

MMI/AMD supplies several software products for its programmable logic devices. Table 2 lists the software supporting the various PLDs.

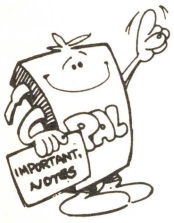
| SOFTWARE | VENDOR | DESCRIPTION |
|---|---|---|
| PALASM 2 (PAL, PROSE, PLS) | MMI/AMD<br>Contact Local Sales Office | Page 3-5<br>Documentation: Chapter 4 |
| PLPL<br>(AmPAL) | MMI/AMD<br>Contact Local Office | Page 3-7 |
| LCA Development Systems (LCA) | MMI/AMD<br>Contact Local Office | Page 3-21 |
| ABEL<br>DASH-GATES<br>DASH-CADAT<br>DASH-ABEL | Data I/O Corp.<br>10525 Willows Road N.E.<br>Redmond, WA. 98073<br>(800) 426-1045 | Page 3-35 |
| CUPL | Personal CAD Systems<br>1290 Parkmoor Avenue<br>San Jose, CA. 95126<br>(408) 971-1300 | Page 3-43 |
| LOG/iC | ISDATA GmbH<br>(Reps: See page 3-75) | Page 3-66 |

**Table 1. Software Design Tools**

| PRODUCT | SOFTWARE SUPPORT |
|---|---|
| 16L8, 16R8, 16R6, 16R4<br>18P8<br>22V10<br>20L10<br>20L8, 20R8, 20R6, 20R4 | Both PALASM 2 and PLPL software* |
| 16RA8<br>16P8, 16RP8, 16RP6, 16RP4<br>10H8, 12H6, 14H4, 16H2, 16C1, 10L8, 12L6, 14L4, 16L2<br>32VX10<br>22RX8<br>20RA10<br>20S10, 20RS10, 20RS8, 20RS4<br>20X10, 20X8, 20X4<br>6L16, 8L14<br>12L10, 14L8, 16L6, 18L4, 20L2, 20C1<br>32R16<br>105, 167, 168<br>14R21<br>10H20G8<br>10H20P8<br>10H/10020EV/EG8 | PALASM 2 software only |
| 23S8<br>29MA16<br>29M16<br>22XP10, 20XRP10, 20XRP8, 20XRP6, 20XRP4<br>22P10, 20RP10, 20RP8, 20RP6, 20RP4 | PLPL Software only |
| 16X4 | PALASM 1 Software |
| 29PL141 | ASM14X |
| 2971 | PEGASUS |
| M2064<br>M2018 | XACT software |

*PALASM 2 and PLPL software are bundled together.

Table 2. Software Support


IMPORTANT
NOTES

# PALASM 2 Logic Design Software Package

## High-Performance Support Tools

PALASM 2 CAD software is an integral part of the MMI/AMD programmable logic solution. As PAL devices and other PLDs have grown more powerful and complex, our team of software engineers has added major enhancements to PALASM software. The goal is to provide timely, state-of-the-art software support for every new PAL device at market introduction. The result is software that enables you to configure a PLD quickly, easily, and effectively.

## Freedom to Express Your Designs in Different Forms

PALASM 2 software offers you increased design flexibility. You have the option of creating your design file with Boolean or State equations. The powerful PAL device design specification syntax has the advantage of being flexible enough for complex designs, without compromising ease-of-use. The basic operators INVERT, AND, OR, and EXCLUSIVE-OR can be used to describe any logic function using Boolean equations. The syntax for State equations is equally easy to use.

## Powerful Simulator Provides Automatic Testing

PALASM 2 software has a powerful, event-driven simulator that cuts down the margin of design error significantly. It enables simulation of the design before the chip is programmed. This means you can go back and edit the design as many times as you want without wasting a single chip. The simulator's English-like commands allow you to describe functions easily. It performs a validation of your design, and generates vectors from a test sequence that you specify. PALASM 2 software's simulation makes testing of the design an integral part of creating the design. This means that every time you insert a PAL device into the programmer, you can be sure it will be accurately programmed.

## Automatic Logic Reduction for Cost-Effective Design

PALASM 2 software gives you the option of automatically reducing your logic equations, enabling you to utilize your PAL device fully. Now you don't have to go through tedious manual reduction and DeMorganization. The software does the work for you. Reduced logic leads to cost-effective design, since less device space is used. By conserving space, design efficiency is increased, as more complex logic can be packed into the device.

## Edit Programmed Device Designs

PALASM 2 software offers you the unique ability to edit programmed device designs. Its time-saving JEDEC manipulator enables you to read a fuseplot directly from a programmed device, and disassembles the fuse information back to Boolean equations. If you wish to alter the design, you can edit the Boolean equations that the JEDEC manipulator generates.

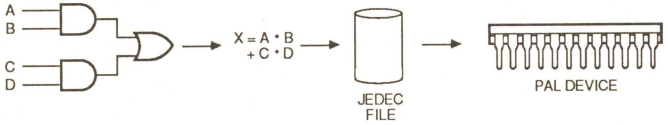
## Easy-to-Use New Menu

The power of PALASM 2 software has been harnessed by a powerful new menu. Function key options allow you to modify, assemble, and simulate your design; view any data, including simulation waveforms; and download JEDEC files to a programmer. And with the "Autorun" feature, all of the assembly and simulation processes can be chained together so that one command completes the entire process. Errors are flagged on-screen and in a log file for examination later. The result is a smooth, integrated design environment that allows you to design logic easily and efficiently.

## Hardware Support

PALASM 2 software is supported on the following systems:

- IBM-PC/XT/AT and compatibles
- VAX-VMS/Ultrix
- Daisy workstations
- Mentor workstations



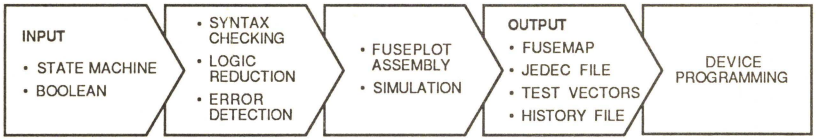

511 01

## Documentation is a Few Pages Away

PALASM 2 software is fully documented in section 4 of this databook. In addition, a free hotline is provided to answer any questions you may have about the software or about MMI/AMD devices. The hotline number is (800) 222-9323.

## Design Software for PLDs

We believe that PALASM 2 software and MMI/AMD PLDs are firmly linked. From immediate device support to documentation to field service: PLD support and software support are one and the same. It is through this philosophy that PALASM 2 software has become the world's most widely-used PLD design package, and a natural complement to MMI/AMD PLDs.




511 02

# PLPL: Programmable Logic Programming Language

## Software Version V2.1

PLPL supports all AmPAL devices, including the following:

16L8, 16R8/6/4 *
18P8 *
23S8
22V10 *
20L10 *
22P10, 20RP10/8/6/4
22XP10, 20XRP10/8/6/4
29M16
29MA16

Devices marked with an asterisk are also supported by PALASM 2 software. PLPL will automatically be shipped with the PC version of PALASM 2 software.

PLPL is a programmable logic development package which lets the designer describe logic functions and state machines in a high-level syntax. Various programs in the PLPL package are used to process this design or source file before programming a device.

PLPL is composed of 6 separate programs:

- A programmable logic compiler (PLC) converts the design file into logic equations and stores these in an intermediate file.
- A logic optimizer (OPTIMIZE) logically reduces the Boolean equations in the intermediate file.
- A JEDEC-standard fuse map generator (JM) converts the equations in the intermediate (or optimized) file and writes these into a fuse map file. This fuse map is used to program the device.
- A manual test vector generator (TESTV) generates JEDEC-standard test vectors from a user-specified function table in the design file. These vectors are used by the simulator when modeling the part.
- A functional simulator (SIM) tests the logic equations in the intermediate/optimized file using the user-defined test vectors.
- A PLD program which helps the user define the architecture features on a device (available for PCs).

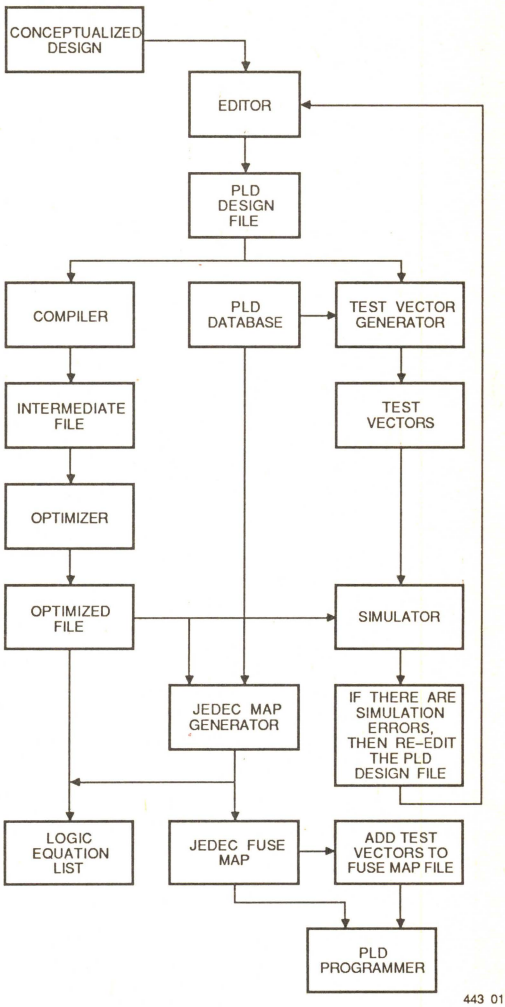
PLPL has a database file for every supported part. Each database file name is composed of the letter P and the numeric designation of the part. For example, the AmPAL22V10 database file is called P22V10.

### PLD Design Methodology: Using PLPL

A typical PLPL design cycle contains the following steps:

1. Write a design file specifying the logic functions to be programmed into a PLD using the PLPL language.
2. Use PLC to compile the design file; the output of PLC is called an intermediate file.
3. If required, use the optimizer to reduce the logic equations in the intermediate file produced by PLC.
4. Specify a function table in the PLD design file. Use TESTV to generate JEDEC-format test vectors from the function table.
5. Use JM to produce a JEDEC-standard fuse map from the equations in the intermediate file.
6. Use SIM to simulate the logic model represented by the intermediate file with the test vectors generated by TESTV.
7. If there are any errors, repeat steps (1) to (6).
8. Load the fuse map into a PLD programmer to program the PLD.

### PLPL PLD Design Environment




443 01

## The PLPL Logic Language

PLPL is a logic language used to simplify the design and definition of Boolean logic functions. These functions can be described using logic equations with Boolean operators in canonical or standard sum-of-products form, or through high-level language constructs such as IF-THEN-ELSE and CASE.

### Language Elements

There are three main elements in the PLPL language: keywords, punctuation marks, and user-defined elements.

### Keywords

The following is the list of keywords that the PLC compiler attributes special meanings to:

| | | | |
|---|---|---|---|
| BEGIN | DEVICE | END | PRESET |
| CASE | ELSE | IF | RESET |
| DEFINE | ENABLE | PIN | THEN |

These should not be used as variable or constant names.

### Punctuation Marks

These symbols are interpreted in the PLC language:

| | | |
|---|---|---|
| + | → | Boolean OR operator, as in C = A+B |
| * | → | Boolean AND operator, as in C = A*B |
| % | → | Boolean XOR operator, as in C = A%B |
| (,) | → | Parentheses to control logic evaluation, as in C = A*(B+E) |
| / | → | Boolean complement operator, as in C = /A |
| = | → | Assignment operator, as in C = A |
| " | → | Encloses comments |
| ; | → | Statement terminator. This must be put at the end of each statement |
| : | → | Indicates a range of values |
| , | → | Concatenates values and variables in CASE statements and functions |
| . | → | Indicates the end of file and must be preceded by the keyword "END" |

Comments must begin and end with double quotes ("). Such comments can be placed anywhere in a PLPL file to improve readability and documentation. Comments cannot be nested.

### Operator Precedence

There are four logical operators in PLC: NOT (/), AND (*), OR (+), and XOR (%). In addition to these, parentheses '()' are provided to control the grouping or associativity of these operators. The operators are arranged in order of precedence as follows:

In the expression F = A*B + /C + D, A and B will be ANDed first because '*' has higher precedence than '+'. C is complemented before the '+' operator is evaluated. That is, the expression is evaluated as F = ((A*B) + ((/C) + D)).

| OPERATION | OPERATOR | ASSOCIATIVITY |
|---|---|---|
| primary | () | right to left |
| bitwise complement | / | right to left |
| bitwise AND | * | right to left |
| bitwise OR,XOR | +,% | right to left |

Note that the '+' and '%' operators have the same precedence. Use parentheses to prevent any ambiguities in the logic expression.

Example: F = (A%B) + (C%D) will be evaluated differently from
F = A%B + C%D

### User-Defined Elements

You can create variables or numbers in PLPL. Variables are alphanumeric strings which begin with an alphabetic character and may contain up to 24 characters. These include all 26 letters, the numbers 0-9, and the underscore ('_') symbol. Spaces cannot be used, and upper and lower-case characters are treated the same.

Example: VAR_A this is a valid variable name
VAR A invalid variable name

Numbers can be expressed in one of four radices: binary, octal, decimal and hexadecimal. To specify a radix, the '#x' symbol is used, where 'x' is b, o, d, or h to represent binary, octal, decimal, or hexadecimal, respectively. If '#x' is not used, the number is assumed to be decimal. In PLC, the numbers have to be positive integers.

| | | |
|---|---|---|
| Example: | #b1110 | binary representation of 14 |
| | #o016 | octal representation of 14 |
| | #d14 | decimal representation of 14 |
| | 14 | decimal representation of 14 |
| | #hE | hexadecimal representation of 14 |

Note: Upper or lower-case characters can be used for the keywords, variables, or numbers. For example, no distinction will be made between the character strings "DEVICE" and "Device".

## PLPL Design File

The logic equations or function definitions are specified in an ASCII PLPL design file. Most text editors/word processors can create clean text files in ASCII mode, which are free of any control characters. The design file contains the following sections: the design name, the header, and the logic specification.

Example:

```
DEVICE design_name (part_name)
                  "<—— Design name"

PIN               "<—— header section"
  D[0] = 1(input combinatorial)
  Y[0] = 22 (output active_HIGH registered);
DEFINE
  CONSTANT_1 = 4,
  CONSTANT_2 = 5; "<—— end of header section"

BEGIN   "<—— logic/function section"
        "write logic equations in this section"


END.
```

### Design Name

The design name section contains the keyword DEVICE, the design file name, and the part to be used in parentheses.

```
DEVICE design_name (part_name)
```

### Header

The header consists of two subsections: pin definition and define sections.

#### 1. Pin Definition Section

The designer defines names and architectural features to each pin on the PLD. For example, the AmPAL16R8 (P16R8) has 8 inputs and 8 registered active-LOW outputs. A design making use of 2 inputs and 5 outputs on this device can be described as follows:

```
DEVICE example (P16R8)
PIN  a = 1 (input combinatorial)
     /b = 2 (input combinatorial) "active_LOW input"
     /state[3:0] = 13:16 (output registered active_LOW)
     /c = 18 (output registered active_LOW);
```

In this example, pins 1 and 2 have been defined as the input variables A and /B (active low inputs have '/'), and pin 18 as the registered output variable C that is also active-LOW.

The architecture definitions in parentheses correspond to the features available for this device pin. If a feature is programmable (e.g., registered or combinatorial on the AmPAL22V10), then these architecture definitions are necessary for the JM fuse map generator program to set the appropriate architecture fuses. The dedicated input pins do not need the "active_LOW" definition in parentheses because there is no physical fuse to program in order to get an active-LOW input; the '/' is sufficient.

To get the available architecture settings for each pin, use the menu-driven PLD program. This program guides the user through the definition of each pin. After setting the architecture fuses, the PLD program will write a PLPL source file template containing the design name and header sections. All the designer has to do is enter the logic equations between the main BEGIN-END section.

Pins can also be associated in groups called vectors. Once a group of pins has been defined as a vector, this group can be referred to by the vector definition. This is helpful when specifying state machines or address/data buses. In the example, the group of pins 13,14,15, and 16 have been assigned to the output vector called "state[3:0]". This is logically equivalent to the definition:

```
PIN  a = 1 (input combinatorial)
     /b = 2 (input combinatorial)
     /state[3] = 13 (output registered active_LOW)
     /state[2] = 14 (output registered active_LOW)
     /state[1] = 15 (output registered active_LOW)
     /state[0] = 16 (output registered active_LOW)
     /c = 18 (output registered active_LOW);
```

The range of pins to be assigned to a vector can be described by using the ':' symbol, as in "13:16". In addition, non-sequential pin numbers can be specified by using ','. In the pin definition example, if pin 17 is to be used instead of pin 16, then the definition of state[3:0] can be written as:

```
/state[3:0] = 13:15,17 (output...)
```

An element or elements in a vector can be accessed by using the appropriate subscripts.

```
Example:  c = a*b*state[3];
              "access the 3rd vector element"
          state[3:2] = state[1:0];
              "assign the last two vector elements
              to the two most significant bits"
```

Only pins with the same architecture definitions can be grouped together as vectors.

#### 2. DEFINE section (optional)

PLPL supports intermediate variable definitions. A PLPL definition is a variable name assigned to an integer constant or an often-used logic equation. Vector definitions are currently not supported. Each macro definition is separated by a comma and the macro definition section is terminated by ';'. This is an optional section.

```
Example: DEFINE LOAD        = ENABLE1*ENABLE2 ,
                OUTPUT1     = 20 ,
                SET_SIGNAL  = LOAD + SYSRESET ;
```

In the example, LOAD has been assigned to the logic equation ENABLE1*ENABLE2, while the name OUTPUT1 has been assigned to the constant decimal 20. The logic equations assigned to defined names can contain variables and logic operators. The

variables can also be previously defined names, as shown in the SET_SIGNAL definition, where the definition LOAD is logically ORed with the signal SYSRESET.

Definitions are used to simplify the logic specification section by assigning easily recognized names to logic equations or constants. It is easier to remember that a load signal is LOAD instead of a logic equation ENABLE1*ENABLE2, and that an often-used value is called OUTPUT1 instead of the decimal number 20.

## Logic/Function Description

After defining the pin architectures, the designer can now write logic equations for these pins between the main BEGIN-END.

### Logic Specification Section

The statements in the main BEGIN-END block that describe the logic functions can be expressed in terms of logic equations or high-level statements.

### Logic Equations

The equations can use the logic operators described in the language elements section. This capability is provided for designers who know the logic equations for a function. The logic equation is composed of three parts: the variable on the left-hand side, the assignment symbol '=', and the logic expression on the right-hand side of the '='. Each logic equation is terminated by a semicolon (;).

### Single Signal Expressions

The left-hand side of the equation can be a pin name or a pin vector. All pins must be defined in the PIN definition section as an output, I/O or internal register. The logic expression on the right-hand side of the '=' can be any Boolean algebraic expression composed of the logic operators AND (*), OR (+), NOT (/), and XOR (%). In addition, evaluation of logic statements can also be controlled by the use of parentheses. Each equation is considered a single statement.

Example: If the variables B and C are to be evaluated first, then these are enclosed in parentheses.

```
C = A*(B+C)*/D ;
```

Every signal in the equation section is considered either true or false regardless of the physical pin description. A statement such as:

```
IF (A*B) THEN
  C = 1; "this is logically equivalent to C = A*B"
```

means output C is true (set to a logic 1) if signals A and B are true. "IF (/A*/B) THEN" is read "IF A and B are both false". Note that the designer does not need to worry about the pins being defined as active HIGH or LOW.

### Vector Expressions

Logic operations can also be performed on vectors. If a vector VCTR_A[3:0] is to be assigned the value of another vector VCTR_B[0:3] logically ANDed with a vector VCTR_C[3:0], then this is written as:

```
VCTR_A[3:0] = VCTR_B[0:3]*VCTR_C[3:0];
```

This is easier to write than:

```
VCTR_A[3] = VCTR_B[0]*VCTR_C[3];
VCTR_A[2] = VCTR_B[1]*VCTR_C[2];
VCTR_A[1] = VCTR_B[2]*VCTR_C[1];
VCTR_A[0] = VCTR_B[3]*VCTR_C[0];
```

Single signals can also be used when working with vector variables.

Example: `VCTR_A[3:0] = VCTR_B[0:3]*/A;`

is equivalent to:

```
VCTR_A[3] = VCTR_B[0]*/A;
VCTR_A[2] = VCTR_B[1]*/A;
VCTR_A[1] = VCTR_B[2]*/A;
VCTR_A[0] = VCTR_B[3]*/A;
```

Vectors cannot be assigned to single signals, as in

```
C = VCTR_A[3:0];
```

This vector assignment property also holds for vectors specified with special functions (e.g., ENABLE, RESET). Vectors cannot be created by concatenating scalars or parts of vectors (using ',') in a logic expression such as "VCTR_A[3:0] = A,B,VCTR_A[1:0];". An error will be generated.

## High-Level Logic Descriptions

The designer can describe logic functions in a higher-level format by making use of the PLPL statement constructs. PLPL supports two statement forms: IF-THEN-ELSE and CASE.

### IF-THEN-ELSE Statement

This language format is similar to the IF-THEN-ELSE used in regular programming languages. In PLPL, this language constructs the appropriate logic equations from the statements in the THEN and ELSE sections and the test conditions. The statement format is:

```
IF (logic condition) THEN
   [statement]
ELSE
   [statement]
```

For example, if an output pin is to be set when a condition (e.g., /A) is true, and reset when not true, this can be defined as:

```
IF (/A) THEN
  OUTPUT = 1;
ELSE
  OUTPUT = 0;
```

This is the same as writing OUTPUT = /A, where OUTPUT will be active when the condition /A is true and inactive when not true. The IF-THEN-ELSE statement makes the function more understandable. A 2-input AND gate can be similarly described:

```
IF (A*B) THEN
  OUTPUT = 1;
ELSE
  OUTPUT = 0;
```

Truth Table

| A | B | OUTPUT |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

The high-level description is equivalent to OUTPUT = A*B; The logic test condition must always be enclosed in parentheses.

Note that the statement following THEN and ELSE can be a single statement or a group of statements enclosed between BEGIN and END, and followed by a semicolon (;). For example:

```
IF (/A) THEN
  BEGIN
  A = B+C;
  G = VCTR_A[3]+ B;
  END;
ELSE
  etc..
```

The entire IF-THEN-ELSE statement is considered a single statement and can be nested inside another IF-THEN-ELSE.

```
IF (/A) THEN
  IF (B+C) THEN "nested IF-THEN-ELSE"
    C = A*B;
  ELSE
    C = A*D;
ELSE
  A = B;
```

The ELSE part in any IF-THEN-ELSE is optional but any ELSE section will match the most recent IF section, hence care must be taken when using nested IF-THEN-ELSE statements.

For example:

```
IF (/A) THEN
  IF (B+C) THEN "nested IF-THEN"
    C = A*B;
ELSE
  A = B;
```

The ELSE is matched with IF (B+C) THEN, and not IF (/A) THEN. In order to match the ELSE with (/A), BEGIN and END keywords must be used to enclose the statements between IF (/A) THEN and the ELSE, as shown in the following example:

```
IF (/A) THEN
  BEGIN
  IF (B+C) THEN "nested IF-THEN is now a single
    statement"
    C = A*B;
  END;
ELSE   "ELSE now matches with IF (/A) THEN"
  A = B;
```

### Logic Test Conditions

A logic test condition can be a logic expression, a vector test, or a combination of both.

### Logic Expression as a Test Condition

A logic expression can be used as a test condition. This expression can contain single signals, vector variables, and logic operators, including parentheses.

Examples:
```
IF (/A) THEN
IF (A+B*(A+/C)) THEN
IF (A+VCTR_A[3]) THEN
IF (VCTR_A[3:0] = #b1001) THEN
```

In the last example, a vector test is used as the test condition. This logic expression checks if the vector is a specific value. The value can be expressed in any radix, as long as it can be represented by the vector. VCTR_A[3:0] is tested to determine if it has a value of binary 1001; this is equivalent to

```
IF (VCTR_A[3]*/VCTR_A[2]*/VCTR_A[1]*VCTR_A[0]) THEN
```

If the vector test condition does not include the equal sign and a value (as in IF (VCTR_A[3:0]) THEN), then this is equivalent to logically ANDing every element in the vector, or

```
IF (VCTR_A[3]*VCTR_A[2]*VCTR_A[1]*VCTR_A[0]) THEN
```

Vector test conditions can be mixed with other vector or single-signal test conditions. The following are some examples of mixed element logic test conditions:

Example:
```
IF ((VCTR_A[3:0] = #b1001)*/a + b) THEN
  .....
IF ((VCTR_A[0:3] = #hA)*(VCTR_B[3:0] = #o12))
  THEN
  .....
```

Note that each vector test must be enclosed in its own set of parentheses. If not, an error will be generated. A vector test condition can be performed by concatenating single signals and vectors and testing for a value. In the example below, signals A,B and VCTR_A[2] are tested to determine if they have the value #b110.

Example:
```
IF (A,B,VCTR_A[2] = #B110) THEN
```

### CASE Statement

The CASE statement is similar to the multiway branch statement provided in computer programming languages. It has the following format:

```
CASE(pin_vector)
 BEGIN
  value0) [statement]
  value1) [statement]
  value2) [statement]
    :
    :
  valueN) [statement]
 END;
```

The pin vector must be large enough to represent the values VALUE0 to VALUEN. For example, if the pin vector contained two elements, then a maximum of four different values can be tested. The user can also specify a range of values by using the ':' and ',' symbols.

Example:
```
CASE (VCTR_A[3:0])
   BEGIN
     0:5,9) BEGIN
            F = A*B;
            E = /A*C + B;
            END;
12,#b1111) A_FLAG = 1;
   END; "end of CASE"
```

In the example above, the CASE statement is used to check the possible values of VCTR_A. The first values tested are from 0 to 5 and the decimal number 9. The second test checks whether VCTR_A[3:0] is equal to 12 or 15 (specified in binary). Any number radix can be used to specify the values, and a name defined as a constant in the DEFINE section can also be used as a CASE value.

The statement at each variable value can be a single logic equation, a set of logic equations (bracketed by BEGIN and END), an IF-THEN-ELSE statement, or another CASE statement. There is no default statement to handle values that are not specified. No logic equations will be generated for the unspecified values.

Example:
```
CASE (VCTR_A[3:0])
BEGIN
  0 ) BEGIN
      VCTR_A[3:0] = 1;
      A_FLAG = 1;
      END;
  1 ) BEGIN
      VCTR_A[3:0] = 12;
      A_FLAG = 0;
      END;
  12) VCTR_A[3:0] = 0;
END; "end of case statement"
```

In the example above, only three possible values for VCTR_A[3:0] are tested and the corresponding logic statement(s) are listed.

The vector used in the CASE condition can also be created from single signals. The concatenation operator ',' is used to group single signals and/or vectors together, as shown in the examples below.

Examples:
```
CASE (A,B,C)
  BEGIN
        #B100) [statement]
           :
  END;

CASE (A,VCTR_A[3:2],C)
  BEGIN
    #B1100) [statement]
         :
  END;

CASE (VCTR_A[3:2],VCTR_B[3:2])
  BEGIN
    #B1101) [statement]
         :
  END;
```

Case statements are useful in creating state machines. The vector variable specified in the CASE statement can be considered as a group of state bits, with the values in the CASE statement being the range of possible states the vector may take.

A multi-mode counter is an example of a state machine. By defining two pins as state registers (these can be defined as a vector), the count sequence can be easily customized. In this example, the next state of the machine at any count is determined by the present count and the mode bit MODE. If the MODE bit is UP, then the counter operates as an up counter. If the MODE bit is DOWN, then the device operates as a down counter.

### Multi-Mode Counter Example

```
DEFINE UP = 0, "constant definitions for better"
       DOWN = 1; "readability"

CASE (COUNT[1:0]) "4-state up/down counter"
  BEGIN
  0) IF (MODE = UP) THEN
       COUNT[1:0] = 1;
     ELSE
       COUNT[1:0] = 3;
   ::::::
  3) IF (MODE = UP) THEN
       COUNT[1:0] = 0;
     ELSE
       COUNT[1:0] = 2;
  END; "end of CASE statement"
```

NOTE: The use of CASE statements with large numbers of values to be tested and the extensive use of nested CASE statements may force the PLC compiler to consume all available memory. If an "OUT OF MEMORY" error is generated when using nested CASE statements, try converting the function definition into a single-level CASE statement.

### DeMorganization

If an output variable or vector is prefaced by a complement (/) operator on the left-hand side of a logic equation, then the right hand side of the logic equation is DeMorganized.

Example:
```
IF (/A) THEN
   /SIGNAL[3:0] = /B*C;
```

In this example, the right-hand side will be DeMorganized before assignment to each element in the vector SIGNAL[3:0]. In other words, the resulting logic expression for each element in the vector SIGNAL is:

```
SIGNAL[3] = B + /C;
SIGNAL[2] = B + /C;
SIGNAL[1] = B + /C;
SIGNAL[0] = B + /C;
```

### Positive-Polarity Signals on Negative-Polarity Outputs

A positive-polarity signal can be represented with a negative-polarity pin by DeMorganizing the equation. This may be necessary if the only available PLD does not have programmable polarity. For example, the active-HIGH function F = A*B can be implemented on an active-LOW device (P16L8) by complementing the function:

```
/F(H) = /(A*B)
 F(L) = /(A*B)
 F(L) = /A + /B
```

In the PLPL language:

```
DEVICE a_design (P16L8)
PIN A = 1 (input combinatorial)
    B = 2 (input combinatorial)
    F = 15 (output active_LOW combinatorial);
BEGIN
/F = A*B;
END.
```

PLPL automatically DeMorganizes the equation for device implementation.

## Special Functions

The PLPL language allows the creation of logic expressions to utilize special functions for PLD pins. RESET, PRESET and ENABLE are examples of special functions for output pins. These three function names are treated as keywords because most PLDs incorporate them. More advanced devices may have other control functions (e.g., OBSERVE on the AmPAL23S8).

The database file contains architecture information on the device that is used to generate the JEDEC fuse maps for programming the device. The database file also contains architecture and special function names which the user will need to know when writing the PLPL source file for the part.

Example:

The polarity architecture fuse for pin 23 on a 22V10 is defined in the database file as:

```
# active_LOW 5808 0 + # active_HIGH 5808 1
```

This means JEDEC fuse number 5808 for the 22V10 controls the polarity of this device. If the name "active_LOW" is used in the pin definition section (described earlier) for pin 23, then fuse 5808 is set to state 0; if "active_HIGH", then the fuse is set to state 1.

Special functions such as OBSERVE and PRELOAD are also contained in the database file. These names are prefaced by the '!' symbol.

Example:

The observability product term feature on the 23S8 is defined in the database file as:

```
!OBSERVE 1 6072;
```

This is interpreted as 1 observability product term starting at JEDEC fuse location 6072. The user must type the feature name when using the special function in the design file.

The special functions for newer PLDs are listed in the corresponding PLD database file and are preceded by the '!' symbol. ENABLE, RESET and PRESET are not listed there. PC users can use the PLD program to get a listing of the functions available on a PLD.

### Usage

A special function consists of the function name followed by parentheses. Enclosed in the parentheses are the signals and/or vectors that are to be associated with the logic expressions used to define and activate the special functions.

For example, to define the special function product term(s) for a set of output vectors, write:

```
DEVICE a_design (P22V10)
PIN MODE0 = 1 (input combinatorial)
    MODE1 = 2 (input combinatorial)
    A = 3 (input combinatorial)
  SIGNAL[3:0] = 14:17 (output registered
    active_HIGH);

BEGIN
IF (/MODE0*/MODE1) THEN
  ENABLE(SIGNAL[3:0]) = #b1111; "i"
IF (MODE0*/MODE1) THEN
  RESET(SIGNAL[3]) = A;    "ii"
IF (MODE0*MODE1) THEN
  PRESET(SIGNAL[3]);     "iii"
        :
END.
```

In (i), the ENABLE function product term for each vector element is set to 1 (logic true) when both mode inputs are low because the binary value #b1111 corresponds to each of the four vector elements. If the binary number #b1 is used, then the enable term for SIGNAL[0] will be set to 1 and the enable terms will be 0 for the other 3 vector elements.

The special functions can also be equated to logic expressions. In (ii), the RESET function is active for the vector element SIGNAL[3] if the test condition (MODE0*/MODE1) is true and the variable A is true. If no logic expression is specified (iii), then the function is dependent on the test condition (MODE0*MODE1).

If no test condition is specified; as in:

```
BEGIN
ENABLE(SIGNAL[3:0]);
...
END.
```

then this is equivalent to ENABLE(SIGNAL[3:0]) = #b1111; where each variable enclosed in the parentheses is assigned the constant 1.

The function can also be defined like a logic expression. (ii) can be written as: "RESET(SIGNAL[3]) = MODE0*/MODE1*A;".

## Special Parts

### Using the XOR Gate in the AmPAL20XRP10 Series

To use the XOR gate in the AmPAL20XRP10 PLD series (e.g., P22XP10), use the XOR function provided:

```
pin20 = input1*input2*input3;
xor(pin20) = /input1*/input2+/input3*/input4;
```

The second statement above will assign the logic function to the two product terms allotted to the XOR gate of pin PIN20. The first statement above assigns the logic expression to the other group of six product terms on the other XOR input. Do not use the XOR (%) operator because the compiler will convert logic expressions with the % operator into its sum-of-products form.

### Internal Registers on the AmPAL23S8

The internal registers on the AmPAL23S8 are considered as output pins and are numbered as pins 21 to 26. Pin 21 refers to the internal register connected to physical output pin 13, while pin 26 is connected to pin 18.

## Using the Dual-Feedback Macrocell on the AmPALC29M/MA16

The architecture of each of the 16 I/O macrocells on the 29M16 and 29MA16 can be configured by the 8 or 9 architecture fuses in each macrocell. 8 of the 16 macrocells on the part have dual-feedback paths and these dual-feedback macrocells have 8 architecture fuses. The remaining 8 single-feedback macrocells have 9 fuses. The 8 dual-feedback macrocells are connected to pins 3,4,9,10,15,16,21, and 22.

In PLPL V2.1, each of the architecture fuses is given a pair of names describing the effect the fuse has on the macrocell. For example, fuse S0 (see datasheet) can be described as "ACTIVE_LOW" or "ACTIVE_HIGH". This corresponds to the 2 possible states this fuse can have. For fuses S4 to S7, the names Sn_0 or Sn_1 are used (n = 4,5,6,7), where Sn_0 means the fuse is not programmed, and Sn_1 means the fuse is programmed.

The fuse names can be obtained from the P29M/MA16 database files. The fuse names are prefaced by the # symbol. For a description of the functions controlled by the architecture fuses, refer to the datasheet.

These names are put between parentheses in the pin definition section in the PLPL design file.

### Using the Dual-Feedback Macrocell as an Internal Register and a Dedicated Input

The dual-feedback macrocell can be used as an internal register and as a dedicated input. This is done by setting architecture fuses in the macrocell to always disable the output buffer. To refer to this internal register in the part, use the virtual pin number.

There are 8 virtual pin numbers corresponding to the 8 dual-feedback macrocells. These virtual pins are numbered from pins 25 to 32, and refer to the physical pins 3, 4, 9, 10, 15, 16, 21, and 22 respectively. For example, if the register attached to pin 3 is programmed as an internal register, then pin 3 can be considered as an input and pin 25 as an internal register. Logic equations can be assigned to pin 25.

Example:
```
DEVICE dual_macro (P29M16)
PIN  p2 = 2 (input combinatorial)
     p3 = 3 (I_O combinatorial)
     p4 = 4 (I_O combinatorial)
     regular_out = 5 (output active_HIGH
                      registered reg_latch
                      out_cell s4_1 s5_1
                      s6_0 s7_1
                      reg_feedback)
internal1 = 25 (active_HIGH registered
               reg_latch S4_1 S5_1 s6_0
               s7_0);

BEGIN
internal1 = p2 & p4;
regular_out = internal1*/p3;
END.
```

The architecture definitions for the pin "internal1" have been selected to always disable the output of the register in this macrocell (S6 and S7 set to 0 = S6_0, S7_0). This frees pin 3 to be used as a dedicated input by using one of the feedback paths in this dual-feedback path macrocell.

Other output pins on the device, such as "regular_out", can now refer to this internal register just like a regular pin. The only difference is that the output of this internal register cannot be observed by the designer at the pins.

### Using the Dual-Feedback Macrocell as a Regular I/O Macrocell

The dual-feedback macrocells can be used as regular I/O pins. Configure the architecture features of the virtual pins such that the output of the register in this dual-feedback macrocell is always enabled. This means that the output of the register will be sent through the corresponding physical pin. The physical pin must be configured as an output or I/O pin.

Example:
```
DEVICE dual_macro (P29M16)
PIN  p2 = 2 (input combinatorial)
     p3 = 3 (IO combinatorial)
     p4 = 4 (IO combinatorial)
regular_out = 5 (output active_HIGH
                 registered reg_latch
                 out_cell s4_1 s5_1 s6_0
                 s7_1 reg_feedback)
internal1 = 25 (active_HIGH registered
               reg_latch S4_1 S5_1 s6_0
               s7_1);

BEGIN
internal1 = p2 & p4;
regular_out = internal1*/regular_out;
END.
```

In this example, the output of "internal1" will be sent through its corresponding physical pin 3. Note that P3 is defined as a combinatorial I/O pin.

### Defining I/O Pins as Inputs

When an I/O pin on a device that does not have polarity control (e.g., pin 15 on the AmPAL20L10) is defined as an input, the polarity keywords (ACTIVE_HIGH and ACTIVE_LOW) must not be used.

For example: Pin 15 on the P20L10 is defined as an active-HIGH input. The pin should be defined as:

```
PIN name_x = 15 (input combinatorial)
```

## Generating Test Vectors

Test vectors are used by PLD programmers or logic simulators to verify that the logic functions defined for a PLD are correct. These vectors describe the inputs to the PLD and the outputs expected from the device after applying these inputs.

In PLPL, these test vectors are listed at the end of the design file. They are processed by the test vector generator program (TESTV) which produces JEDEC-format test vectors.

Test Vector Format

```
DEVICE ....
PIN ....  See PLPL Language section
BEGIN

END.

TEST_VECTORS
[Pin Classification]
BEGIN
[Vectors]
END.
```

The user-defined test vectors are attached to the end of a PLPL language file, i.e. after the "END.". The keyword TEST_VECTORS marks the beginning of the vector section. This is followed by a pin classification section which specifies the pin types. There are four pin types: IN, OUT, I_O, and BREG. These refer to input, output, input/output and internal (buried) register pins, respectively.

The pin names specified in the pin classification section must have been defined already in the PIN definition section (see PLC). The pin names must also be classified under the appropriate types.

Example:
```
DEVICE Ex1 (Pxxxxx)
PIN A = 1 (input ...)
   /B = 2 (input ...)
   /C = 15 (IO active_LOW...)
    D = 16 (output ...);
BEGIN  "logic equation section"
  ::::
END.
TEST_VECTORS  "test vector section"
IN A,B;
I_O C;
OUT D;
BEGIN
  :::::
END.
```

The pins A and B are classified as inputs, C as I/O, and D as a dedicated output pin. This matches with the PIN section. Note that the '/' symbols can be used, but are not required.

The pin classification section specifies the order with which the user must specify the pin values. This means that since the pin order is now A,B,C and D, the values specified for the test vector must also follow this order.

Example:
```
'TEST_VECTORS
IN A,B;
I_O C;
OUT D;
BEGIN
1 1 H L;
1 0 H H;
0 1 L L;
0 0 L H;
END.
```

In the first vector, the first value 1 is associated with pin A, the next 1 with pin B, H with pin C, and L with pin D. The list is then terminated with a ';'.

If a different pin order is required, then the pin order in the pin classification section can be changed.

Example:
```
TEST_VECTORS
OUT D;
IN A;
I_O C;
IN B; "sections can be split up also"
BEGIN
L 1 H 1;
::::
END.
```

## Test Vector Values

The values a pin can take in a test vector are determined by its pin type. These values are outlined in the JEDEC standard. The following is the list of possible pin values:

- 0 — drive input low
- 1 — drive input high
- 2-9 — drive input to super voltage #2 to 9
- L — test output low
- H — test output high
- F — float input or output
- Z — test input or output for high-impedance
- C — drive input low, high, low (positive clock pulse)
- K — drive input high, low, high (negative clock pulse)
- P — preload registers
- B — preload buried/internal registers
- N — power pins and output not tested
- X — output not tested, use input default level

The default level for unspecified pins is a 0 or L. In the test vector generator program TESTV, this can be set to a 1 or H.

## Pin Types

There are two types of pins: supply, and input or output pins.

### Supply Pins

These pins are not tested by the PLD programmer or logic simulator. These are the power and ground pins. They should not be specified in the pin classification section.

### Input and Output Pins

Input and Output pins are dedicated input, output, clock, or input/output pins. Control pins such as dedicated enable pins (as on the AmPAL16R8) are considered input pins. The values these pins may take are listed below:

| PIN TYPE | POSSIBLE TEST VECTOR VALUE |
|---|---|
| input | 0,1,2,3,4,5,6,7,8,9<br>F,Z<br>X,N |
| output | L,H,F,Z<br>0,1(preload)<br>X,N |
| clock | C,K,P<br>X,N |
| input/output | (same as input and output) |

Example: To test a 2-input AND function programmed into an AmPAL16R8 with the following PLPL definition:

```
DEVICE AND_FUNCTION (P16R8)
PIN    CLK1 = 1 (clock)
       A = 2 (input combinatorial)
       B = 3 (input combinatorial)
       ENB = 11 (control) "enable pin"
      /AND = 19 (registered output active_LOW);

BEGIN
AND = A*B;
END.
TEST_VECTORS
IN CLK1,ENB,A,B; "pin classification section"
OUT AND;
BEGIN
C 0 0 0 L;
C 0 0 1 L;
C 0 1 0 L;
C 0 1 1 H;
END.
```

The JEDEC test vectors produced will be:

```
      V0001 C00XXXXXXN0XXXXXXXHN*
      V0002 C01XXXXXXN0XXXXXXXHN*
      V0003 C10XXXXXXN0XXXXXXXHN*
      V0004 C11XXXXXXN0XXXXXXXLN*
            ^^^      ^       ^
   Pin      1        11      19
             2
              3
```

Note that the values for pin 19 are inverted. This is because the name AND in the pin classification section did not have a '/' as in the PIN definition section. TESTV reverses the polarity of the vector value if the pins do not have the same definitions (one defined with the '/' and the other without). If they have the same definition in both the pin classification and definition section (both with or without '/'), the vector values are not modified.

This capability is useful because now the user can think in terms of asserted/not asserted or voltage levels. If the user is thinking of assertion levels, then the names are specified in the pin classification section without any '/'s. A 0/1 or L/H will mean not asserted/asserted for inputs and outputs, respectively. TESTV will convert the 0/1s and L/Hs by resolving any polarity discrepancies in both the pin classification and definition sections.

Example: In the AND function example, pin 19 (AND) is defined as active-LOW in the PIN definition section but defined without the '/' in the pin classification section. The vector values specified now refer to AND being asserted (H) or not asserted (L). TESTV will automatically invert them.

If the user wants to think in terms of voltages with a 0/1 and L/H referring to the low and high voltages respectively, then the names must match in both pin definition sections.

Example: In the AND example, if pin AND were defined as /AND in the pin classification section, then the test value specified for AND must be the voltage or physical level expected. This means that when AND is asserted, an L is expected at the output, with H expected when AND is not asserted.

The test vectors will be numbered in increasing order (decimal) and will contain a number of values equivalent to the physical number of pins on the device. In the preceding example, all pin locations on the AmPAL16R8 except pins 1,2,3,10,11,19 and 20 are specified as X or don't care. Pins 10 and 20 are the power and GND pins (automatically set by the TESTV program) while pins 1,2,3,11 and 19 were taken from the test vector specification.

Some PLD programming systems and simulators will use the test vectors in the following manner: the specified inputs are applied to the PLD and the outputs and input/output values are compared with the outputs specified. In sequential designs the previous register state is used to determine the next state. For the very first vector, many PLDs have the power-up reset feature which guarantees the part starting with all registers reset to 0.

### Preloading Registers

Registers can be preloaded by putting the 'P' value on the clock pin controlling the registers. This means that if a clock pin controls a bank of registers, a 'P' should be placed on the clock pin to preload that register bank. The value to be loaded into the registers is then specified using 0 and 1, not L and H.

Example: To preload an AmPAL16R8 with 10010110, the following vector sequence is used (note that the outputs are active LOW):

```
TEST_VECTORS
IN CLK1,IN_A,IN_B;
OUT  /out7,/out6,/out5,/out4,/out3,/out2,/out1,/out0;
BEGIN
C 0 1 XXXX XXXX;    "do one test"
P X X 0110 1001;    "preload reg with 10010110"
0 X X LHHL HLLH;    "test registers without clocking"
C 1 1 XXXX XXXX;    "do another test with preloaded reg"
END.
```

For preloading buried or internal registers on a device like the AmPAL23S8, the keyword LOAD_INTERNAL is used. A sequence of 0s and 1s follows which corresponds directly to the states of the internal registers.

1. Test vector values may be grouped together if they are all numbers or all alphabetic characters.

Example:
```
TEST_VECTORS
    IN IN0,IN1,IN2;
    OUT OUT0,OUT1,OUT2;
    BEGIN
    111 HHL;
    100 LHH;
    101 LHL;
    END.
```

2. Spaces can be used to separate the test vector values (ex. '1 1 1 H H L').

## PLPL V2.1 Programs

The PLPL V2.1 package contains the following programs:

Programs:

1. PLC: generic logic software compiler
2. OPTIMIZE: logic optimizer
3. JM: JEDEC map generator
4. TESTV: manual test vector generator
5. SIM: functional simulator
6. For PC users, a pin utility program PLD.EXE is included

A menu program PLPL.EXE can be used to control execution of each of the programs, or the programs can be called individually as described later. Experienced users can create a batch file to perform the COMPILE-OPTIMIZE-FUSEMAP sequence.

### PLC: Programmable Logic Compiler

The PLC logic compiler compiles an ASCII logic description file written in the PLPL language and produces an intermediate file. To run PLC, type the following command at the system prompt (e.g.,A>)

```
A> plc -i filename [-o intermediate_filename]
```

-i filename ==> specifies the input filename
-o intermediate_filename ==> writes the compiled form to the file specified (optional)

The compiled form will appear on the screen. Any errors will be written to the temporary file $tmp.$$$.

### OPTIMIZE: Logic Optimizer

OPTIMIZE is a logic equation optimizer that applies logic reduction algorithms to the expressions in the intermediate file. This program detects and eliminates logic redundancies in the expressions.

The optimizer is called as follows:

```
A> optimize -i intermediate_filename
  [-o new_file]
```

-i intermediate_filename ==> an intermediate file is taken as an input

-o new_filename ==> the logically minimized file is written to the file with the specified name (optional)

The optimized file will be displayed on the monitor.

### JM: JEDEC Link/Fuse Map and Equation Listing Generator

JM takes an intermediate file, and generates the link/fuse map for the targeted PLD. The link/fuse map generated conforms to the standards set forth by the JEDEC committee. The intermediate file contains the logic equations that are then converted into a pattern of 1s and 0s corresponding to the device links on a PLD.

JM can also perform the following functions:

- List the logic equations with the user-defined signal names into a file; improves documentation
- Concatenate an existing JEDEC map with a TESTV-generated test vector file; this format is used by some PLD programming units with testing capability
- Generate a new PLD design file header containing a correct pin definition section; this menu-driven function will display the available architectural features for all the pins on a user-specified PLD. This is used on VAX versions of the program and is similar to PLD.EXE on the PC.

To run JM, type:

```
A>  jm -i intermediate_filename  [-o map_filename]
                                 [-l list_filename]
        or
    jm -a <link/fuse_map_file> <test_vector_file>
        or
    jm -n
```

-i intermediate_filename ==> generate the link/fuse map from the equations in this file
-o map_filename ==> send the JEDEC map to the file specified (optional)
-l list_filename ==> list the logic equations with the user-specified variables into a file (optional)
-a <link/fuse_map_file> <test_vector_file> ==> first file name contains link/fuse map and second contains the JEDEC standard test vector file
-n ==> used to generate a new PLD design file header

The device/fuse map will be displayed on the screen. If the "-o" option is used, it will also be sent to the file specified.

The "-a" or concatenate option must be used by itself. If other options like "-i" or "-o" are included with their arguments, they will be ignored.

The "-n" option must also be used by itself when creating new design file headers.

### TESTV: JEDEC Standard Test Vector Generator

TESTV takes a PLPL language file and searches for a test vector section. It then converts these user-specified vectors into a format that can be loaded into a PLD programmer for testing.

To run TESTV, type:

```
A> testv -i filename [-o out_filename]
```

-i filename is the input file specification
-o filename is the file to write the test vectors to; this is an optional argument

The vectors generated will be sent to the screen.

## SIM: Functional Simulator

The functional simulator SIM simulates PLD designs created with the PLPL V 2.1 package. Logical errors can be detected before the part is actually programmed, thus reducing debugging costs and design time.

To run the simulator, type:

```
sim -i <intermediate/optimized_file>
          <testvector_file>
```

The following simulator options are available:

```
-o output_file
-b init_val,final_val
-x value_of_don't_care
-z value_of_three_state
-e
-t
```

Options:

```
-e
```

Status information (e.g., number of errors) generated by the Simulator is normally sent to the CRT. This option will cause the output to the CRT to be suppressed.

```
-i    <intermediate_file> <testvector_file>
```

The intermediate file is generated by the PLC program. This file serves as the simulation model for the PLD design. The test vector file is generated by TESTV and contains the inputs to the model and the expected outputs.

```
-o    <output_file>
```

Writes the simulation results to the output file.

```
-b    <init_val,final_val>
```

Breakpoint selection option: simulation will be performed on a range of vectors. This range begins at the vector numbered INIT_VAL up to and including the vector numbered FINAL_VAL. Any one of these two values can be left out. If the initial value is not specified, then the beginning of the test vector file is taken as the first vector read by the simulator. If the final_value is not specified, then the final vector in the test vector file is taken as the final vector affected by this breakpoint option.

For example:

[-b 20,] start simulation at test vector 20 to the end of file.
[-b ,12] start simulation at test vector 1 to test vector 12.
[-b 7,24] start simulation at test vector 7 to test vector 24.

-x <value_of_don't_care>

The default interpretation of a don't care symbol in the test vector file ('X') is interpreted 0 (or L for outputs). The user can set the X value to be interpreted as 0 (L for outputs) or 1 (H for outputs).

```
-z <value_of_three_state>
```

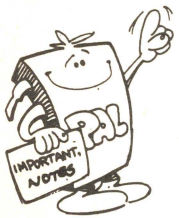
The default three-state value is 1. This Z value can be set to 0, L, or H with this option.

```
-t
```

The trace feature displays simulation results on the screen. The simulator will compare the calculated outputs with the expected outputs (specified by the user) and flag any inconsistencies.

## Simulating Special Functions

To ensure the correct simulation of a design specification, make sure that all special-function product terms for outputs on a PLD are defined. If a product term is not used, specify a logic 0 for that product term. For example, if the RESET function is not used on an AmPAL22V10, write: RESET(x,y) = #b00; where x,y are two outputs used in the design.


IMPORTANT
NOTES

# Logic Cell™ Array and Development Systems

## The Logic Cell Array

The Logic Cell Array (LCA) is the first device to successfully bridge the gap between field programmable logic and gate arrays. The LCA™ successfully combines the benefits of low-power CMOS LSI technology and the advantages of user programmability with the gate density and logic flexibility previously obtainable only with gate arrays.

The LCA provides a quantum jump in field-programmable logic device capability extending its usable functional density into a realm beyond that of more conventional programmable logic devices. Much greater gate utilization is achieved with the LCA by use of a flexible array type architecture more versatile than that of conventional PLDs, which is increasingly inefficient as gate density is increased. The Monolithic Memories M2018 1800-gate LCA device can replace as many as six 1200-gate PLD devices in some applications.

Gate arrays, on the other hand, provide densities higher than those of current LCAs. However, gate arrays typically require longer development times, design risks and significant cost.

The LCA is the ideal option for the PLD designer wishing to achieve a new level of system functional density and for the gate array user looking for a low-cost and easy-to-use alternative which provides instant prototyping through the power of in-circuit emulation

## Component Ordering Information

M2018-50 CNL84

PART NUMBER
M2064 (1200 Gates, 58 IOB)
M2018 (1800 Gates, 74 IOB)

SPEED GRADE
-33 = 33 MHz Toggle Rate
-50 = 50 MHz Toggle Rate
-70 = 70 MHz Toggle Rate

PACKAGE TYPE
N48 = 48 Pin Molded DIP
NL68 = 68 Pin PLCC
NL84 = 84 Pin PLCC
P68 = 68 Pin PGA
P84 = 84 Pin PGA

TEMPERATURE RANGE
C = Commercial
M = Military

## Package Availability

| PART NUMBER | 48-PIN PLASTIC DIP N48 | 68-PIN PLCC NL68 | 68-PIN PGA P68 | 84-PIN PLCC NL84 | 84-PIN PGA P84 |
|---|---|---|---|---|---|
| M2064 | X | X | X | | |
| M2018 | | X | X | X | X |

## Ordering Information Development Systems

| PART NUMBER | DESCRIPTION |
|---|---|
| LCA-MEK01 | Logic Cell Array Evaluation Kit |
| LCA-MDS21 | XACT™ Design Editor System |
| LCA-MDS22 | P-SILOS™ Simulator |
| LCA-MDS23 | Automatic Placement and Routing Program |
| LCA-MDS24-48N | XACTOR™ In-Circuit Emulator for 48-Pin DIP (includes one LCA-MDS26 and one LCA-MDS27-48N) |
| LCA-MDS24-68NL | XACTOR In-Circuit Emulator for 68-Pin PLCC (includes one LCA-MDS26 and one LCA-MDS27-68NL) |
| LCA-MDS24-68P | XACTOR In-Circuit Emulator for 68-Pin PGA (includes one LCA-MDS26 and one LCA-MDS27-68P) |
| LCA-MDS24-84NL | XACTOR In-Circuit Emulator for 84-Pin PLCC (includes one LCA-MDS26 and one LCA-MDS27-84NL) |
| LCA-MDS24-84P | XACTOR In-Circuit Emulator for 84-Pin PGA (includes one LCA-MDS26 and one LCA-MDS27-84P) |
| LCA-MDS26 | Universal Emulation Pod |
| LCA-MDS27-48N | Emulation Header Cable for 48-Pin DIP |
| LCA-MDS27-68NL | Emulation Header Cable for 68-Pin PLCC |
| LCA-MDS27-68P | Emulation Header Cable for 68-Pin PGA |
| LCA-MDS27-84NL | Emulation Header Cable for 84-Pin PLCC |
| LCA-MDS27-84P | Emulation Header Cable for 84-Pin PGA |
| LCA-MDS31 | FutureNet® DASH™ Schematic Design Entry Interface |

## Service Contracts

| PART NUMBER | DESCRIPTION |
|---|---|
| LCA-MSC21 | XACT Design Editor System (LCA-MDS21) Annual Support Agreement |

## Logic Cell Array M2064, M2018

### Features

- Fully Programmable
  - I/O functions
  - Digital logic functions
  - Interconnections
- General purpose array architecture
- Complete user control of design cycle
- Compatible arrays with logic cell complexity equivalent to 1200 and 1800 usable gates
- Standard product availability
- 100% factory-tested
- Selectable configuration modes
- Low-power, CMOS, static memory technology
- Three performance options: 33, 50, and 70 MHz
- TTL or CMOS input threshold levels
- Complete development system support
  - XACT Design Editor
  - Macro Library
  - Timing analyzer
  - Design rules checker
  - Configuration file generator
  - Configuration file formatter
- Optional features
  - Schematic capture entry
  - XACTOR in-circuit emulator
  - Logic and timing simulator
  - Auto Place/Route

### Description

The Logic Cell Array (LCA) is a high-density CMOS user-programmable logic device. The array architecture of the LCA allows the designer total flexibility and yields extremely high gate utilization. The LCA is composed of three configurable logic elements: Input/Output Blocks (IOBs), Configurable Logic Blocks (CLBs), and Programmable Interconnect. The XACT development system Design Editor provides a graphical interface to configure individual IOBs for external interface, define CLBs to implement internal logic, and assemble an internal network of interconnect to accomplish larger logic functions. The XACT Design Editor provides an interactive graphic design capture system with an automatic routing feature. Both logic simulation and emulation are available for design verification.

### Programming

The Logic Cell Array's logic functions and interconnections are determined by a configuration program stored in internal static memory cells. On-chip logic provides for automatic loading of configuration data at power-up or on command. The program data can reside in an EEPROM, EPROM, or ROM on the circuit board or on a floppy disk or hard disk.

Several methods of automatically loading the required data are designed into the Logic Cell Array and are determined by logic levels applied to mode selection pins at configuration time. The form of the data may be either serial or parallel, depending on the configuration mode. The programming data are independent of the configuration mode selected.

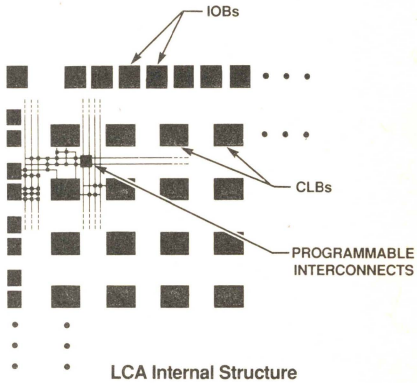
The Logic Cell Array is available in a variety of logic capacities, package styles, temperature ranges and speed grades.

#### Input/Output Block

Each user-configurable I/O block (IOB) provides an interface between the external package pin of the device and the internal logic. Each I/O block includes programmable input path and a programmable output buffer as shown in Figure 1. It also provides input clamping diodes to provide protection from electrostatic damage, and circuits to protect the LCA from latch-up due to input currents.

The input buffer portion of each I/O block provides threshold detection to translate external signals applied to the package pin to internal logic levels. The input buffer threshold of the I/O blocks can be programmed to be compatible with either TTL (1.4 V) or CMOS (2.2 V) levels.

Output buffers in the I/O blocks provide 4-mA drive for high fan-out CMOS- or TTL-compatible signal levels.




LCA Internal Structure

| PART NUMBER | LOGIC CAPACITY (USABLE GATES) | CONFIGURABLE LOGIC BLOCKS | USER I/Os | CONFIGURATION PROGRAM (BITS) |
|---|---|---|---|---|
| M2064 | 1200 | 64 | 58 | 12038 |
| M2018 | 1800 | 100 | 74 | 17878 |

### Configurable Logic Block

An array of Configurable Logic Blocks (CLBs) provides the functional elements from which the user's logic is constructed. The Logic Blocks are arranged in a matrix in the center of the device. The M2064 has 64 such blocks arranged in an 8-row by 8-column matrix. The M2018 has 100 logic blocks arranged in a 10 by 10 matrix.

Each logic block has a combinatorial logic section, a storage element, and an internal routing and control section as shown in Figure 2. Each CLB has four general-purpose inputs: A, B, C, and D; and a special clock input (K), which may be driven from the interconnect adjacent to the block. Each CLB also has two outputs, X and Y, which may drive interconnect networks.

Additional memory bits are used to set the user-definable path selectors, shown in Figure 2, which determine CLB internal connections. All memory bits are determined automatically by the XACT design editor as the design is entered.

The logic block combinatorial logic uses a table look-up memory to implement Boolean functions. This technique can generate any logic function of up to four variables with a high-speed, sixteen-bit memory. The propagation delay through the combinatorial network is independent of the function generated. Each block can perform any function of four input variables or any two functions of three input variables each. The input variables may be selected from among the four inputs and the block's storage element output "Q."

### Programmable Interconnect

Programmable interconnection resources in the Logic Cell Array provide routing paths to connect inputs and outputs of the I/O and logic blocks into desired networks. All interconnections are composed of metal segments, with programmable switching points provided to implement the necessary routing. Three types of resources accommodate different types of networks:

- General-purpose interconnect
- Long lines
- Direct connection

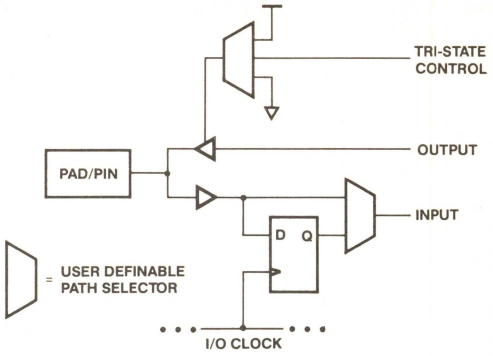



Figure 1. IOB Logic Equivalent

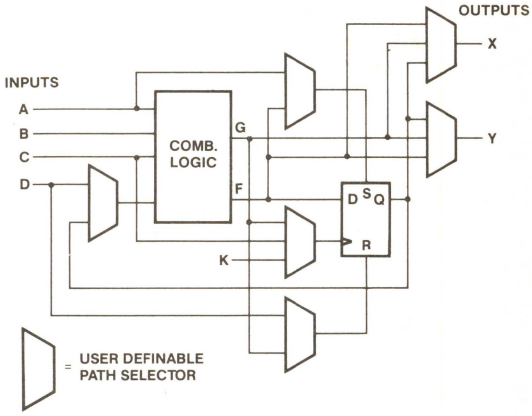



Figure 2. CLB Logic Equivalent

## Summary of CLB Switching Characteristics

| SYMBOL | PARAMETER | | SPEED GRADE | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | -33 | | -50 | | -70 | | |
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{ILO}$ | Logic input to output | Combinatorial | | 20 | | 15 | | 10 | ns |
| $t_{CKO}$ | K Clock | To output | | 20 | | 15 | | 10 | ns |
| $t_{ICK}$ | | Logic-input setup | 12 | | 8 | | 7 | | |
| $t_{CKI}$ | | Logic-input hold | 0 | | 0 | | 0 | | |
| $t_{PID}$ | Input/Output | Pad to input (direct) | | 12 | | 8 | | 6 | ns |
| $t_{OP}$ | | Output to pad (enabled) | | 15 | | 12 | | 9 | ns |
| $F_{CLK}$ | Maximum flip-flop toggle frequency | | 33 | | 50 | | 70 | | MHz |

## LCA-MDS21 XACT Design Editor System

### Features

- Runs on an IBM® PC-XT™ or compatible computer
- Complete basic system for designing with Logic Cell Arrays
- Interactive graphical design editor
- Simplified definition, placement and interconnection capability for logic design and implementation
- Macro library of 113 standard logic family equivalents
- Utility for user-defined macros
- Boolean equation or Karnaugh map alternatives to specify logic functions
- Point-to-point timing calculations for critical path analysis
- Automatic design consistency checking for connectivity and design violations
- Documentation support with hardcopy output of logic and physical configuration information
- Download cable to transfer configuration programs from personal computer to LCA in target system
- Compatible hardware and software options to enhance design productivity
- File formatter for EPROM programmer

### General

The XACT Design Editor provides users with a complete design and development system for specification and implementation of designs using Monolithic Memories' Logic Cell Arrays. Functional definition of Configurable Logic Blocks (CLBs), Input/Output Blocks (IOBs) and interconnection is performed with a menu-driven interactive graphics editor. An automatic router greatly reduces the effort to interconnect logic.

Designs are captured with a graphics-based design editor using either a mouse for menu-driven entry, or a keyboard for command-driven entry. Functions are specified by CLB and IOB definitions plus their interconnections. The macro library and user-defined macros enable the user to easily implement complex functions.

The check for logic connectivity and design rule violation is easily performed. All unused internal nodes are automatically configured to minimize power dissipation.

Interactive point-to-point timing delay calculation is provided for timing analysis and critical path determination. This ability enables the user to quickly identify and correct timing problems while the design is in progress.

Automatic generation of similar input netlist files with timing parameters simplifies the use of P-SILOS for logic and timing simulation.

The XACT Design Editor includes hardcopy generation to document a design and automatically track design changes. Logic Cell Array configuration programs can be automatically translated into standard EPROM programming bit pattern formats.

A download cable included with XACT is useful for transferring configuration programs serially from the PC workstation to a Logic Cell Array installed in a system. During product development and debug this capability can be used to save the time required to write a modified configuration program into an EPROM.

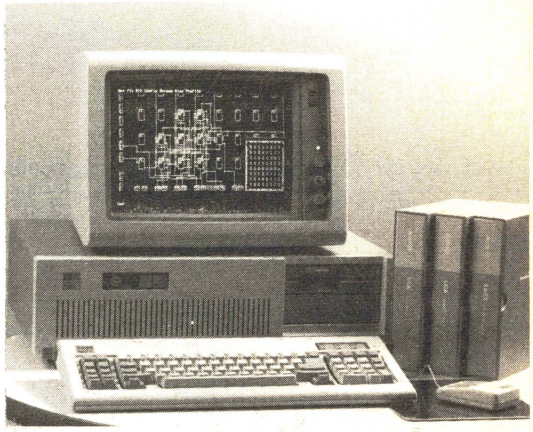
Monolithic Memories provides ongoing support for XACT users. For the first year, software updates are included. After that the user may purchase the LCA-MSC21 Annual Support Agreement to continue to receive the latest software releases. XACT users also receive Monolithic Memories' technical information, which includes information about Logic Cell Arrays and PAL® devices, as well as software updates and application notes for designers. In addition, Monolithic Memories provides comprehensive field and factory support.

### System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer with:

- MS-DOS™ 2.1 or higher
- 1M Bytes RAM
- 1 Diskette Drive
- 10-MB Hard Disk
- IBM compatible Color Graphic Adapter and Display
- 1 Serial Interface Port
- 1 Parallel Interface Port
- Mouse System™, Microsoft® or compatible mouse



Design Editor with Routed Design

# XACT Macro Library

| General | | CLBs |
|---|---|---|
| GADD | Adder | 1 |
| GCOMP | Compare | 1 |
| GEQGT | Equal or Greater | 1 |
| GMAJ | Majority | 1 |
| GMux | 2-to-1 Mux | 1 |
| GPAR | Parity | 1 |
| GXOR | Exclusive-OR | 1 |
| GXOR2 | Dual Exclusive-OR | 1 |
| GXTL | Crystal Oscillator | 0 + 2IOB |
| GOSC | Low Frequency Resistor-Capacitor Oscilator | 1 + 2IOB |

| Pads | | IOBs |
|---|---|---|
| PIN | Input Pad | 1 |
| PINQ | Input Pad with Storage | 1 |
| PIO | Input/Output Pad | 1 |
| PIOQ | Input/Output Pad with Input Storage | 1 |
| PIOC | Input/Output Pad with 'Open Collector' | 1 |
| PIOQC | Input/Output Pad with Input Storage, 'Open Collector' | 1 |
| POUT | Output Pad | 1 |
| POUTC | Output Pad with 'Open Collector' | 1 |
| POUTZ | Output Pad with 3-State Control | 1 |
| PREG | Output Pad with Input Storage | 1 |

| Latches | | CLBs |
|---|---|---|
| LD | Data Latch | 1 |
| LC-rd | Data Latch with ResetDir | 1 |
| LC-sd | Data Latch with SetDir | 1 |
| LD-srd | Data Latch with SetDir, ResetDir | 1 |
| LDM | Data Latch with 2-Input Data Mux | 1 |
| LDM-rd | Data Latch with 2-Input Data Mux, ResetDir | 1 |
| LDM-sd | Data Latch with 2-Input Data Mux, SetDir | 1 |

| Flip-Flops | | CLBs |
|---|---|---|
| FD | D Flip-Flop | 1 |
| FDR | D Flip-Flop with Reset | 1 |
| FDS | D Flip-Flop with Set | 1 |
| FD-rd | D Flip-Flop with ResetDir | 1 |
| FD-sd | D Flip-Flop with SetDir | 1 |
| FD-srd | D Flip-Flop with SetDir, ResetDir | 1 |
| FDC | D Flip-Flop with ClkEna | 1 |
| FDCR | D Flip-Flop with ClkEna, Reset | 1 |
| FDCS | D Flip-Flop with ClkEna, Set | 1 |
| FDM | D Flip-Flop 2-Input Data Mux | 1 |
| FDMR | D Flip-Flop 2-Input Data Mux, Reset | 1 |
| FDMS | D Flip-Flop 2-Input Data Mux, Set | 1 |
| FDM-rd | D Flip-Flop 2-Input Data Mux, ResetDir | 1 |
| FDM-sd | D Flip-Flop 2-Input Data Mux, SetDir | 1 |
| FSR | Set-Reset Flip-Flop with Set Dominate | 1 |
| FRS | Set-Reset Flip-Flop with Reset Dominate | 1 |
| FJK | J-K Flip Flop | 1 |
| FJKS | J-K Flip Flop with Synchronous Set | 1 |
| FJK-rd | J-K (Set-Reset) Flip Flop with ResetDir | 1 |
| FJK-sd | J-K (Set-Reset) Flip Flop with SetDir | 1 |
| FJK-srd | J-K (Set-Reset) Flip Flop with SetDir, ResetDir | 1 |
| FT0 | Self Toggle Flip-Flop | 1 |
| FT0R | Self Toggle Flip-Flop with Reset | 1 |
| FT | Toggle Flip-Flop | 1 |
| FTP | Toggle Flip-Flop with ParEna | 1 |
| FTP-rd | Toggle Flip-Flop with ParEna, ResetDir | 1 |
| FTR | Toggle Flip-Flop with Reset | 1 |
| FTS | Toggle Flip-Flop with Set | 1 |
| FT2 | 2-Input Toggle Flip-Flop | 1 |
| FT2R | 2-Input Toggle Flip-Flop with Reset | 1 |

| Decoders | | CLBs |
|---|---|---|
| D2-4 | 1-of-4 Decoder | 2 |
| D2-4E | 1-of-4 Decoder, with Ena | 2 |
| 74-139 | 1-of-4 Single Decoder with Low Output, Ena | 4 |
| D3-8 | 1-of-8 Decoder | 5 |
| D3-8E | 1-of-8 Decoder with Ena | 6 |
| 74-138 | 1-of-8 Decoder with Enables, Low Output | 7 |
| 74-42 | 1-of-10 Decoder with Low Output | 8 |

| Multiplexers | | CLBs |
|---|---|---|
| M3-1 | 3-to-1 Mux | 2 |
| M3-1E | 3-to-1 Mux with Ena | 2 |
| M4-1 | 4-to-1 Mux | 3 |
| M4-1E | 4-to-1 Mux with Ena | 3 |
| 74-352 | 4-to-1 Mux with Low Output, Ena | 3 |
| M8-1 | 8-to-1 Mux | 7 |
| M8-1E | 8-to-1 Mux with Ena | 7 |
| 74-151 | 8-to-1 Mux with Ena Complementary Outputs | 7 |
| 74-152 | 8-to-1 Mux with Low Output | 7 |

# XACT Macro Library

## Registers

| | | CLBs |
|---|---|---|
| **Data Registers** | | |
| RD4 | 4-Bit Data Register | 4 |
| RD8 | 8-Bit Data Register | 8 |
| RE8CR | 8-Bit Data Register with ClkEna, Reset | 8 |
| **Serial to Parallel** | | |
| RS4 | 4-Bit Shift Register | 4 |
| 74-195 | 4-Bit Serial to Parallel Shift Register with ParEna, Reset | 5 |
| 74-194 | 4-Bit Bidirectional Shift Register with ClkEna, ParEna, ResetDir | 12 |
| RS8 | 8-Bit Shift Register | 8 |
| RS8CR | 8-Bit Shift Register with ClkEna, Reset | 8 |
| RS8PR | 8-Bit Shift Register with ParEna, Reset | 8 |
| RS8R | 8-Bit Shift Register with Reset | 8 |
| 74-164 | 8-Bit Serial to Parallel Shift Register with ResetDir | 8 |

## Counters

| | | CLBs |
|---|---|---|
| **Modulo 2** | | |
| C2BCR | 1-Bit Binary Counters with ClkEna, Reset | 1 |
| C2BC-rd | 1-Bit Binary Counters with ClkEna, ResetDir | 1 |
| C2BP | 1-Bit Binary Counters with ParEna | 1 |
| C2BR | 1-Bit Binary Counters with Reset | 1 |
| C2B-rd | 1-Bit Binary Counters with ResetDir | 1 |
| **Modulo 4** | | |
| C4BCP | 2-Bit Binary Counters with ClkEna, ParEna | 3 |
| C4BCR | 2-Bit Binary Counters with ClkEna, Reset | 2 |
| C4BC-rd | 2-Bit Binary Counters with ClkEna, ResetDir | 2 |
| C4JCR | 2-Bit Johnson Counters with ClkEna, Reset | 2 |
| **Modulo 6** | | |
| C6JCR | 3-Bit Johnson Counter with ClkEna, Reset | 3 |
| **Modulo 8** | | |
| C8BCP | 3-Bit Binary Counters with ClkEna, ParEna | 5 |
| C8BCR | 3-Bit Binary Counters with ClkEna, Reset | 4 |
| C8BC-rd | 3-Bit Binary Counters with ClkEna, ResetDir | 4 |
| C8JCR | 3-Bit Johnson Counters with ClkEna, Reset | 4 |
| **Modulo 10** | | |
| C10BC-rd | 4-Bit BCD Counter with ClkEna, ResetDir | 4 |
| C10BCP-rd | 4-Bit BCD Counter with ClkEna, ParEna, ResetDir | 7 |
| 74-160 | 4-Bit BCD Counter with ClkEna, ParEna, ResetDir | 8 |
| C10BP-rd | 4-Bit BCD Counter with ParEna, ResetDir | 6 |
| C10JCR | 5-Bit Johnson Counter with ClkEna, Reset | 5 |
| **Modulo 12** | | |
| C12JCR | 6-Bit Johnson Counter with ClkEna, Reset | 6 |
| **Modulo 16** | | |
| C16BA-rd | 4-Bit Binary Ripple Counter with ResetDir | 4 |
| C16BC-rd | 4-Bit Binary Counter with ClkEna, ResetDir | 4 |
| C16BCPR | 4-Bit Binary Counter with ClkEna, ParEna, Reset | 10 |
| C16BCP-rd | 4-Bit Binary Counter with ClkEna, ParEna, ResetDir | 6 |
| 74-161 | 4-Bit Binary Counter with ResetDir | 8 |
| C16BP-rd | 4-Bit Binary Counter with ParEna, ResetDir | 5 |
| C16BUD-rd | 4-Bit Binary Up-Down Counter with ParEna, ResetDir | 8 |
| C16JCR | 8-Bit Johnson Counter with ClkEna, Reset | 8 |
| **Modulo 256** | | |
| C256FC-rd | 8-Bit Modulo 256 Feedback Shift Register with ClkEna, ResetDir | 9 |

## LCA-MDS22 P-SILOS Simulator

### Features

- **Event-driven logic and timing simulator**
- **Logic network input automatically generated by XACT Design Editor**
- **Control and observation of any physical circuit node**
- **Multiple file input for vectors and commands**
- **Interactive or batch mode operation**
- **Output available in printed or tabular formats**
- **Runs on an IBM PC-XT, PC-AT or compatible personal computer**

### General

P-SILOS is a powerful PC-based simulator that provides event-driven logic and timing simulation of Logic Cell Array designs. Simulation is particularly useful for testing logic or logic segments as well as for verifying critical timing over worst case power supply, temperature and process conditions.

Simulation is useful in several stages of the design cycle. After design entry, simulation may be used to debug logic in an unplaced and unrouted design. This saves design time because logic errors can be detected and corrected prior to final placement and routing. After a circuit has been placed, routed, and then fully debugged using in-circuit emulation, worst case timing may be verified. This enables the user to select the correct Logic Cell Array speed for a particular application.

Network inputs for Logic Cell Array designs are automatically created by the Simgen utility in the XACT system. The network includes logic and routing delay parameters and setup and hold times based upon the selected speed grade operating under worst case conditions. Simulation stimuli are created with a set of clock statements or with an input pattern for either pad inputs or internal nodes. Simulation results are available in tabular, plotted, and graphic formats. This flexibility makes debugging easy for both the circuit function and timing.

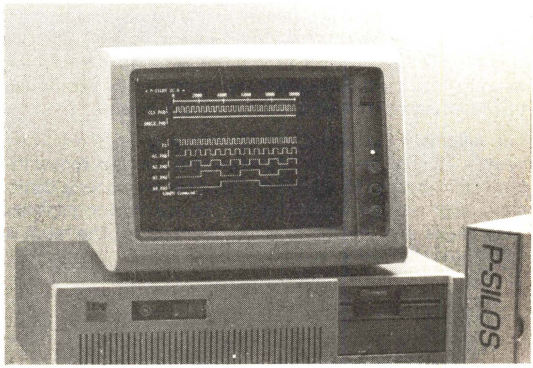
### System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer with:

- MS-DOS 2.1 or higher
- 640 K Bytes RAM
- 1 Diskette Drive
- 10-MB Hard Disk
- 1 Parallel Interface Port

Refer to the MDS21 XACT Design Editor Product Datasheet for additional equipment required for systems which will also run the XACT Design Editor.




**P-SILOS Waveform Output**

## LCA-MDS23 Automatic Placement and Routing Program

### Features

- **Automatic placement and routing of logic to minimize design cycle time**
- **User control over placement of logic blocks**
- **User specification of critical paths**
- **Netlist inputs from either schematic capture or XACT**
- **May be used in conjunction with schematic capture or with the XACT Design Editor**
- **Runs on IBM PC-XT, PC-AT or compatible personal computer**

### General

The automatic Placement and Routing program enhances the productivity of designers using Logic Cell Arrays by reducing design placement and routing time, whether the design logic is entered from a schematic capture package or from the XACT Design Editor.

Designs that are developed incrementally can also take advantage of Automatic Placement and Routing. Partial Logic Cell Array layouts can be locked in place while additions to the design are automatically placed and routed, or the design can be completely rearranged to yield a new placement.

The Automatic Placement and Routing program is extremely flexible. Through placement directives the user can control the placement process to achieve the best placement for a particular design. Routing resources can be specified to minimize clock skews and signal delays for critical paths. The result is faster product developement.

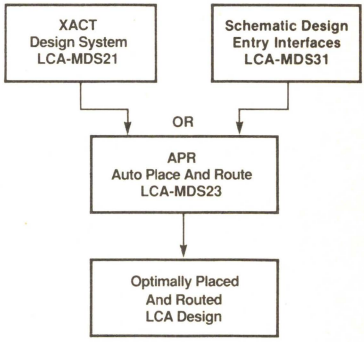
### System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer with:

- MS-DOS 2.1 or higher
- 640 K Bytes RAM
- 1 Diskette Drive
- 10-MB Hard Disk
- 1 Parallel Interface Port

Refer to the MDS21 XACT Design Editor Product Datasheet for additional equipment required for systems which will also run the XACT Design Editor.




**APR Diagram**

## LCA-MDS31 FutureNet DASH Schematic Design Entry Interface

### Features

- **Design entry to XACT via the FutureNet DASH Schematic Designer**
- **Macro library of over 100 standard logic family equivalents derived from the XACT Macro Library**
- **Library of logic symbols including all two-input, three-input, and four-input AND, OR, and XOR gates plus storage, input/output, and clock elements**
- **User control for flagging critical paths for the LCA-MDS23 Automatic Placement and Routing Program**
- **Automatic partitioning and conversion of schematic drawings to a Monolithic Memories' Logic Cell Array design file**
- **Output compatibility with XACT Design Editor and the Automatic Placement and Routing Program**
- **Runs on an IBM PC-XT, PC-AT or compatible personal computer**

### General

Schematic entry and automatic partitioning of Logic Cell Array designs shortens product development times. Complex designs can be specified schematically and quickly implemented for in-circuit design verification.

Monolithic Memories FutureNet DASH Schematic Design Entry Interface provides the symbol library and conversion utility to permit designers to enter Logic Cell Array designs with the FutureNet DASH Schematic Designer. The Monolithic Memories module provides the logic, I/O and macro symbols to be used in the schematic and a conversion utility which automatically partitions and translates the schematic into a Logic Cell Array design.

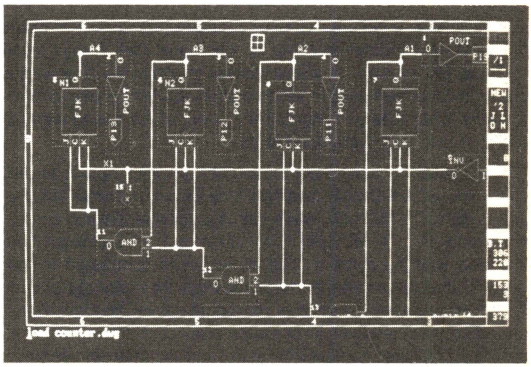
### System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer with:

- FutureNet DASH-2 or later, and associated hardware including mouse, Enhanced Graphics Adapter and Display
- MS-DOS 2.1 or higher
- 640 K Bytes RAM
- 1 Diskette Drive
- 10-MB Hard Disk

Refer to the MDS21 XACT Design Editor Product Datasheet for additional equipment required for systems which will also run the XACT Design Editor.




Schematic Capture

## LCA-MDS24, LCA-MDS26, LCA-MDS27 XACTOR In-Circuit Emulator

### Features

- Real-time in-circuit emulation in user's target system
- Concurrent emulation of up to four devices
- Readback and display of Logic Cell Array internal storage element states
- Device status display with automatic update of asynchronous events
- Control and I/O pin isolation from target system
- Support for daisy chain programming of up to seven devices in a daisy chain
- On-chip crystal oscillator support during emulation
- Support for multiple device and package types
- Runs on an IBM PC-XT, PC-AT or compatible personal computer

### General

The XACTOR real-time in-circuit emulator provides interactive target-system emulation of up to four Logic Cell Arrays from the host PC system. In-circuit emulation provides a powerful productivity enhancement to simulation, providing capabilities to verify functionality in the target system at full speed with all other circuits and system software.

The emulation system is composed of a microcomputer-based controller (LCA-MDS24), and from one to four universal emulation pods (LCA-MDS26), each with a package-specific emulation header (LCA-MDS27). One universal emulation pod is included with the system. The controller is connected to the host PC through a serial port and provides local storage of configuration programs, control of individual device configurations and control of the isolation of the pod device(s) from the target system. The user can set the state and isolation for each of the control signals to provide debugging of target hardware. Four general I/O pins are available to provide test points which may also be isolated from the target system.

Target Logic Cell Arrays can be programmed individually or in a daisy chain. Daisy chains of up to seven devices may be supported from any of the four pods. Individual device isolation and configuration is controlled with mouse or keyboard commands and may be supplemented with user-defined setup files for easy system debugging.

Readback of device configuration may be performed on command for verification of the configuration process and interrogation of the internal states. The state of all internal storage elements is displayed after readback has been performed. Status displays showing the state of all isolation switches and control signal states are provided. The status display includes automatic reporting of asynchronous status changes in the target system.

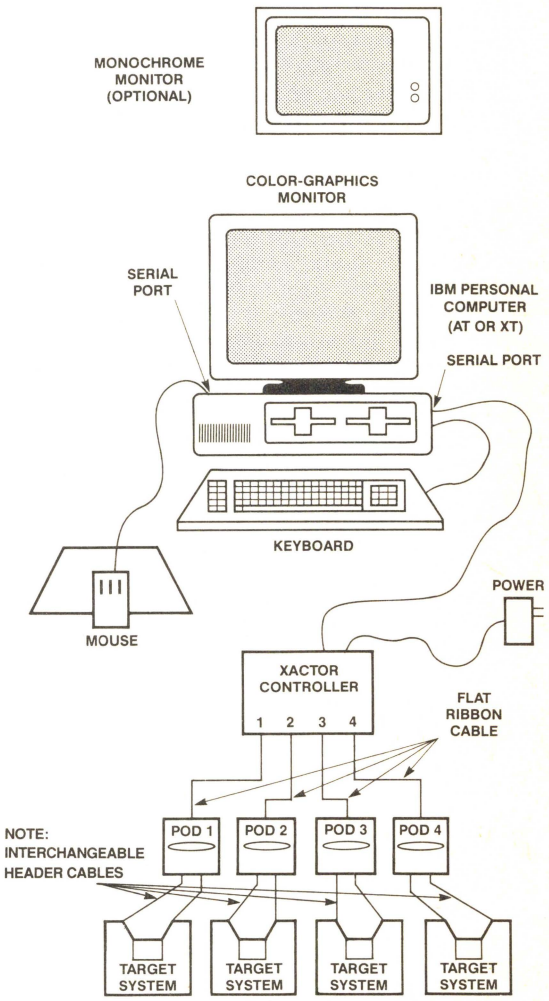
### Universal In-Circuit Emulator Pod (LCA-MDS26)

Additional pods may be connected to the XACTOR in-circuit emulator controller, up to a maximum of four pods per controller. Pod headers (LCA-MDS27) are interchangeable for different device and package types. Each pod provides a direct in-socket connection for a minimum disruption of the target system. Test points are provided to allow connection of a logic analyzer or other test equipment to aid in the system debugging.

### System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer as configured for MDS21 XACT Design Editor, plus second serial interface port.




XACTOR Hardware

# LCA-MEK01 Logic Cell Array Evaluation Kit

The Monolithic Memories Logic Cell Array is a high-performance CMOS user-programmable gate array. The Monolithic Memories' Logic Cell Array Evaluation Kit is a software package that provides the capability to evaluate the Logic Cell Array for new applications.

## Features

- **Design software package for IBM PC-XT, PC-AT or compatible computer**
- **Interactive graphics-oriented designer interface**
- **Simplified definition, placement and connection capability for implementation of complex logic**
- **Boolean equation or Karnaugh map alternatives to specify logic functions**
- **Macro library of 113 standard logic equivalents plus support for user-defined macros**
- **Point-to-point timing calculations for critical path analysis**
- **Automatic checking for connectivity and design consistency**
- **Hardcopy output of logical and physical configuration information**

## General

The Evaluation Kit can be used to enter complete designs using a subset of the XACT design editor, including the use of the Monolithic Memories macro library. Critical timing for the design can be evaluated with the timing delay calculator to evaluate the applicability of the Logic Cell Array technology to a particular design.

Functional definition of Configurable Logic Blocks (CLBs), and their internal routing,I/O Block (IOB) definitions, and interconnection are all done within an integrated graphics-oriented system. Interactive placement and automatic routing of logic and I/O elements are accomplished quickly and easily via an easy-to-learn user interface.

Designs are captured with a graphics-oriented design editor, using either a mouse or keyboard entry, driven from command or files. User functions are specified in terms of CLB definitions and interconnections. Standard logic functions from the macro library or user-defined macro capabilities can be utilized to quickly implement complex logic functions. Placement and routing can be edited easily to modify or optimize a design.

Checking of logical connectivity is performed automatically. All unused internal nodes are automatically configured to minimize power dissipation.

Interactive point-to-point timing delay calculation is provided to simplify timing analysis and critical path determination.

The Evaluation Kit includes hardcopy generation to document a design and automatically track design changes.

## System Requirements

**Minimum System Configuration**

IBM PC-XT, PC-AT or compatible computer with:

- MS-DOS 2.1 or higher
- 640K Bytes RAM
- 1 Diskette Drive
- 10-MB Hard Disk
- IBM or compatible Color Graphic Adapter and Display
- 1 Serial Interface Port
- Mouse Systems, Microsoft or compatible mouse




**Evaluation Kit**

3

## Minimum Requirements of Software and Hardware Configurations for Monolithic Memories LCA Design System

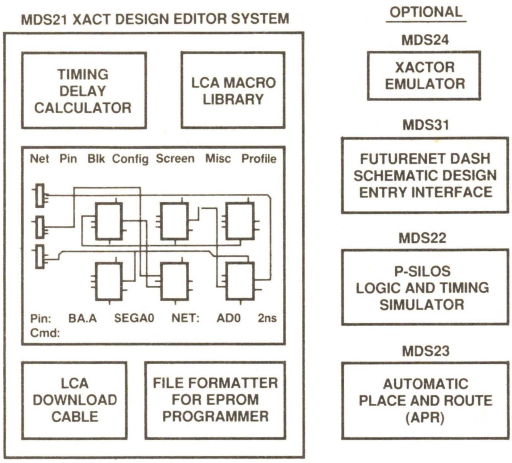
| Legend<br>R Required<br>S Supported<br>— Not required or supported | | Software Package | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | XACT DESIGN EDITOR LCA-MDS21 | | XACT EVALUATION KIT LCA-MEK01 | | P-SILOS SIMULATOR LCA-MDS22 | | AUTOMATIC PLACEMENT AND ROUTING LCA-MDS23 | FutureNet DASH SCHEMATIC DESIGN ENTRY INTERFACE LCA-MDS31 | | | XACTOR IN-CIRCUIT EMULATOR LCA-MDS24 | |
| | | Version 1.30 | | Version 1.30 | | Version 1U.3 | Version 2C.5 | Version 1.0 | Version 1.00** | | | 1.10 | 1.30<br>1.33* |
| | | Version 1.2 | | Version 1.2 | | | | | DASH 2 | DASH 3C | DASH 4 | | |
| XACT Version Required | | | | | | Version 1.2 | Version 1.2 or 1.3 | Version 1.3 | Version 1.3 | | | Version 1.2 | Version 1.3 |
| MS-DOS PC-DOS Operating System | | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | Version 2.1 or above | | | Version 2.1 or above | Version 2.1 or above |
| Logic Cell Arrays Supported | | M2064 (8x8) only | M2064 (8x8) and M2018 (10x10) | M2064 (8x8) only | M2064 (8x8) and M2018 (10x10) | M2064 (8x8) only | M2064 (8x8) and M2018 (10x10) | M2064 (8x8) and M2018 (10x10) | M2064 (8x8) and M2018 (10x10) | | | M2064 (8x8) 68NL only | M2064* (8x8) and M2018 (10x10) |
| IBM PC XT or 100% compatible | | R | R | R | R | R | R | R | R | | | R | R |
| IBM PC AT or 100% compatible | | S | S | S | S | S | S | S | S | | | S | S |
| Memory | Minimum System RAM | 640 KB | 1 MB | 640 KB | 1 MB | 640 KB | 640 KB | 640 KB | 256 KB | 512 KB | 512 KB | 640 KB | 1 MB |
| | Hard Disk (10 MB min 30 MB REC) | R | R | R | R | R | R | R | R | R | R | R | R |
| Graphics Boards and Displays | Monochrome | — | — | — | — | R | R | — | R | — | — | — | — |
| | CGA (Color graphics adapter) | R | R | R | R | S | S | R | — | — | — | R | R |
| | EGA (Enhanced color graphics adapter) | S | S | S | S | S | S | S | — | R (with 192 KB) | R (with 192 KB) | S | S |
| | Lotus/Intel EMS (Expanded memory specifications) | — | R (with 256 KB) | — | R (with 256 KB) | — | — | — | — | — | — | — | R (with 256 KB) |
| | Vendor graphics board | | | | | | | | Future-Net graphics controller board | | | | |
| Other Devices | Security key | R | R | — | — | R | R | R | — | — | — | R | R |
| | Mouse | R† | R† | R† | R† | — | — | — | R (Future-Net) | R (Future-Net) | R (Future-Net) | R† | R† |
| | Min. number of parallel ports | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Future-Net mouse/ parallel port board | Future-Net mouse/ parallel port board | 1 | 1 |
| | Min. number of serial ports | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 |

* XACTOR Version 1.33 supports the universal emulator pod with interchangeable header cables for each package type. Versions 1.10 and 1.30 or XACTOR support only the dedicated 68-Pin PLCC emulation pod originaly offered with XACTOR Version 1.10. XACTOR Version 1.33 will also support the original 68-Pin PLCC Emulator Pod.

** LCA-MDS31 FutureNet DASH Schematic Design Entry Interface version 1.00 is compatible with FutureNet DASH Schematic Designer versions 2, 3C and 4.

† Must be Mouse Systems™, FutureNet® or Microsoft® mouse compatible.

## Logic Cell Array




**Development System**

The DS21 XACT Design Editor provides all capabilities required for Logic Cell Array design. Additional development system options provide enhanced designer productivity during design entry, placement and routing, and design verification.

Xilinx, Logic Cell, XACT, XACTOR and LCA are trademarks of Xilinx, Inc.

IBM is a registred trademark and PC, PC/AT, PC/XT are trademarks of International Business Machines Corporation.

FutureNet is a registered trademark and DASH is a trademark of FutureNet Corporation, a Data I/O Company. P-Silos is a trademark of SimuCad Corporation. MS-DOS is a trademark of Microsoft Corporation. Mouse Systems is a trademark of Mouse Systems Corporation. Microsoft is a registered trademark of Microsoft Corporation.

Monolithic Memories does not assume any liability arising out of the application or use of any product described herein; nor does it convey any license under its patent, copyright or maskwork rights of any rights of others. Monolithic Memories reserves the right to make changes, at any time, in order to improve reliability, function or design and to supply the best product possible. Monolithic Memories cannot assume responsibility for the use of any circuitry described other than circuitry entirely embodied in their product. No other circuit patent licenses are implied.

Monolithic Memories cannot assume responsibility for any circuits shown or represent that they are free from patent infringement or any other third party right.

Monolithic Memories assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made.

Portions of this data sheet reproduced with the permission of XILINX, Inc.


IMPORTANT
NOTES

# ABEL–GATES
## Two Powerful Tools For PLD Design

### From FutureNet (Data I/O)

FutureNet® Corporation (a Data I/O® Company) currently offers two high-level PLD design tools: ABEL™, and the more sophisticated DASH-GATES™. Each of these products is ideally suited for certain tasks, hardware platforms, and budgets. This article describes the similarities and differences between ABEL and DASH-GATES, and provides brief examples of the uses of each.

### Natural Design Descriptions

No matter how a design is described, all current PLD programming technology adheres to one standard: the JEDEC file (Standard 3A). The JEDEC file contains a list of 1's and 0's that specifies the binary state of each fuse in the PLD. Unfortunately, a JEDEC fusemap is not the way most engineers would like to describe a design; Boolean equations, truth tables and state diagrams are all preferable methods.

Thus, ABEL and DASH-GATES share a common purpose—to generate a JEDEC file from a design description that is more familiar to an engineer. ABEL and DASH-GATES incorporate high-level design languages that help engineers describe designs in the most natural way. The following types of descriptions can be used*:

- Schematics
- Boolean Equations
- Truth Tables
- State Diagrams

These formats can be used in any combination, to describe any design; the engineer is free to decide which form best suits the task at hand. Figure 1 shows a state diagram described with the ABEL state diagram syntax.

** With the addition of FutureNet's DASH-ABEL™, designs can be described in schematic form with the FutureNet DASH™ CAE system and converted to ABEL for implementation in PLDs.*

```
State AddCard:   AddClk    = !ClkIN;
                 Ace       := Ace;
                 if (is_Ace & !Ace) then Add_10 else Wait;

State Add_10:    AddClk    = !ClkIN;
                 Ace       := High;
                 goto       Wait;

State Wait:      AddClk    = Low;
                 Ace       := Ace;
                 if (CardOut==Low) then Test_17 else Wait;

State Test_17:   AddClk    = Low;
                 Ace       := Ace;
                 if !GT16 then ShowHit else Test_22;

State Test_22:   AddClk    = Low;
                 Ace       := Ace;
                 case       LT22          : ShowStand;
                           !LT22 & !Ace    : ShowBust;
                           !LT22 &  Ace    : Sub_10;
                 endcase;

State Sub_10:    AddClk    = !ClkIN;
                 Ace       := Low;
                 goto       Test_17;
```

**Figure 1. ABEL and DASH-GATES Let Engineers Describe Designs at a High Level, as Shown in This Portion of an ABEL State Diagram**

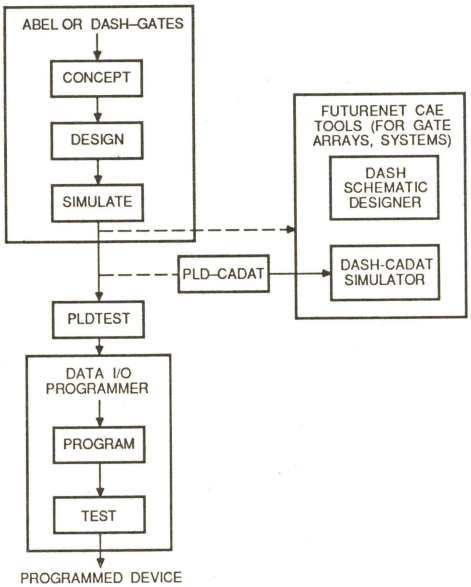
## Simulating and Optimizing the Design

ABEL and DASH-GATES share another major feature: both employ logic reduction algorithms to automatically reduce a design description to a near-minimal form. When entering designs, engineers need not perform reductions themselves using tedious manual methods such as Karnaugh maps. DASH-GATES also factors designs to make equations fit the architecture of the device.

Functional simulation is also performed by ABEL and DASH-GATES. Functional simulation verifies that a design operates as intended before a PLD is programmed. This not only saves PLDs; it also provides an opportunity for the engineer to experiment with new design ideas or changes. Since simulation is automatic and generally takes just seconds or minutes, fast answers to "what-if" questions can be obtained.

### PLD Toolkit

Figure 2 shows how Data I/O and FutureNet tools are used to fully automate the design process. This PLD design toolkit not only enhances each step of the PLD design process, but it also provides a link to system level simulation and gate array implementation.




593 02

### PLDtest™: Test Vector Generation and Fault Analysis

To ensure comprehensive testing of each programmed device, PLDtest analyzes the PLD design description and generates a set of test vectors based on both the design and the target device. PLDtest attempts to assure 100% testability, but reports the actual testability along with a fault analysis if 100% testability cannot be achieved.

### PLD-CADAT™: A Link to System-Level Simulation

The ABEL and DASH-GATES simulators readily perform functional simulation of single PLD designs. PLD-CADAT goes a step further by providing a link to FutureNet's DASH-CADAT-PLUS™ system-level simulator. DASH-CADAT-PLUS can simulate complete boards, providing results of functional simulation, timing analysis, and fault simulation. PLD-CADAT converts JEDEC files created by ABEL or DASH-GATES into the CADAT model description language so that PLD designs can be simulated as part of a much larger CADAT circuit.

## ABEL

First introduced in 1983, ABEL is a high-level design language for PLDs. The early version of ABEL supported 90 devices, and was a full-fledged PLD design tool with logic reduction, simulation, and automatic generation of design documentation.

Today, ABEL supports over 600 devices, providing complete support for virtually all available PLDs. Logic reduction and simulation algorithms have been improved, and language modifications have been made. The newest enhancements to ABEL are:

**Greater Device Support**—ABEL supports virtually all standard PLDs and can call device specific programs (DSPs) to support non-standard devices.

**Simulation**—A new simulator provides greater support of asynchronous devices and complex macrocells. It also allows changes in test vectors without reprocessing of the design.

**Improved Syntax**—The ABEL syntax now supports devices with multiple feedback paths, and is compatible with the DASH-GATES syntax.

**Macro and Function Library**—Device-specific declarations or any kind of macro or function can now be stored in a system library.

**Automatic JEDEC-to-ABEL Conversion**—This utility converts a JEDEC file to an ABEL source file. The JEDEC file might be obtained from disk or directly from a programmed device. This utility is useful for recovering undocumented designs existing on master devices, or for making quick changes to designs for which the ABEL source file has been lost.

ABEL was designed to provide a comprehensive PLD design tool that would perform on standard IBM PCs, XTs, and ATs. ABEL is also available for VAX VMS and UNIX installations, and runs on most popular engineering workstations.

## DASH-GATES

DASH-GATES offers all the PLD design features of ABEL and much more. DASH-GATES provides superior assistance in the design entry process, with split-screen capability, design entry forms, and interactivity.

Additionally, DASH-GATES is a link to full CAE systems (such as FutureNet DASH) which can be used to create random logic, gate array, standard cell, and full system-level designs. Such designs can then be simulated at a system level with complete timing analysis and fault grading.

## The Differences Between ABEL and DASH-GATES

ABEL and DASH-GATES are both powerful PLD design tools, but there are differences between them that make each better-suited for a particular application or class of users. The major differences are outlined below; additional differences are discussed in ABEL and DASH-GATES examples that follow.

### Interactive vs. Batch

The single biggest difference between the two products is that DASH-GATES is a truly interactive program and ABEL is not. DASH-GATES provides forms to speed up design entry and continuously monitors design input to detect errors. As you make an error, DASH-GATES lets you know so you can correct it "on the fly." DASH-GATES provides other "interactive advantages" illustrated in the design examples below. ABEL, on the other hand, processes a design description that has been created with a text editor, reports errors to the screen during processing, and writes detailed error messages to documentation files.

### Links to Other Technologies

ABEL is specifically a PLD design tool and supports virtually all available PROMs, PAL devices, PLSs, and other PLDs. DASH-GATES supports all the same devices as ABEL and provides an automated path to gate arrays and standard cells. DASH-GATES has a functional-to-schematic description conversion feature that automatically converts equations, truth tables, and state diagrams to FutureNet DASH schematics. These schematics can then be incorporated into larger designs using the DASH CAE system, and netlists can be produced for gate array and standard cell designs.

### Advanced Features of DASH-GATES

DASH-GATES also provides factoring, partitioning, and advanced reduction algorithms. These features allow greater flexibility in adapting, or "fitting", a design to meet the constraints of various types of devices.

### Required Hardware

ABEL runs on minimum configuration PCs, including laptops, as well as any IBM PC, XT, AT or compatible personal computers, workstations, or minicomputers. DASH-GATES runs on an enhanced AT, UNIX-based minicomputers and a variety of engineering workstations.

## Describing And Processing Designs

### How a Design is Described with ABEL

ABEL design descriptions are entered using any standard text editor. Any combination of Boolean equations, truth tables, and state diagrams can be used to describe the desired logic function. The basic Boolean operators for ABEL are the following:

```
!  invert
&  AND
#  OR
```

Figures 2 and 3 show a complete design for a counter/seven-segment display decoder. (This same design will be used to illustrate the operation of DASH-GATES.) The design is actually described in two separate modules; each module describes a partition of the design that will be programmed into a device.

The design description has four major sections, as shown in the figures. The declarations section defines set names and assigns signal names to device pins. Sets are useful for referring to a group of signals with one name. Subsequent equations can use the set name in lieu of listing all the components of the set. "Count" is a set of the signals Q0, Q1, Q2, and Q3.

The equations section lists the Boolean equations for the design. In this example, a complete state machine is described with one equation. The equation, "Count = (Count +1) & !Clear", describes the count-up operation that takes place only when Clear is low.

The truth table section of the ABEL design description contains the decoding function for the seven-segment decoder. For each value of count (values can be entered in decimal, binary, hex, or octal), the corresponding outputs for the LED segments are listed. The outputs are expressed in terms of ON or OFF according to the desired state of the LED. ON and OFF can be assigned to a high or low signal based on polarity required to drive the LEDs.

The final section of the design description contains test vectors used to perform functional simulation. Much like the truth table, the test vectors describe inputs and their corresponding outputs. During simulation, the inputs are applied to the design and results are checked against the listed outputs. If a mismatch between actual and predicted values occurs, a simulation error is reported.

### How a Design is Processed by ABEL

An ABEL design is processed in six steps that can be run by issuing one batch file command. Typically, the user types a command like "ABEL LED" to process the design named LED. ABEL then automatically performs the logic reduction, simulation, and conversion to a JEDEC file. Individual steps such as logic reduction or simulation can be performed if desired. Each step of the ABEL processing sequence can be customized through the use of parameters.

### How a DASH-GATES Design is Entered

DASH-GATES uses design entry forms to help the engineer enter PLD designs. The following forms are available:

```
module _count   flag '-r3'

title '4 bit binary counter  FutureNet a Data I/O Company'

        count   device  'P16R8';

        Clk,Clear,OE1           pin 1,2,11;
        Q0,Q1,Q2,Q3             pin 14,15,16,17;

        Count   =  [Q3,Q2,Q1,Q0];

        Z,C     = .Z., .C.;

equations
        Count := (Count +1) & !Clear;

test_vectors
       ([Clk,Clear,OE1] -> Count)
        [ C ,  1  , 0 ] ->    0;
        [ C ,  0  , 0 ] ->    1;
        [ C ,  0  , 0 ] ->    2;
        [ C ,  0  , 0 ] ->    3;
        [ C ,  0  , 0 ] ->    4;
        [ C ,  0  , 0 ] ->    5;
        [ C ,  0  , 0 ] ->    6;
        [ C ,  0  , 0 ] ->    7;
        [ C ,  0  , 1 ] ->    Z;
        [ C ,  0  , 1 ] ->    Z;
        [ C ,  0  , 0 ] ->   10;
        [ C ,  0  , 0 ] ->   11;
        [ C ,  0  , 0 ] ->   12;
        [ C ,  0  , 0 ] ->   13;
        [ C ,  0  , 0 ] ->   14;
        [ C ,  0  , 0 ] ->   15;
        [ C ,  0  , 0 ] ->    0;
        [ C ,  0  , 0 ] ->    1;
        [ C ,  1  , 0 ] ->    0;

end
```

Figure 2. ABEL Source Files for a Counter

```
module _led  flag '-r3'

Title  '7 segment decoder   FutureNet a Data I/O Company'

        led     device  'P16L8';

        Q0,Q1,Q2,Q3     pin 2,3,4,5;
        OE2             pin 11;
        a,b,c,d,e,f,g   pin 13,14,15,16,17,18,19;

        Count   = [Q3,Q2,Q1,Q0];
        ON      = 1;
        OFF     = 0;
        X,Z     = .X., .Z.;

equations
        enable [a,b,c,d,e,f,g] = !OE2;

truth_table
(Count -> [ a , b , c , d , e , f , g ])
   0    -> [ON ,ON ,ON ,ON ,ON ,ON ,OFF];
   1    -> [OFF,ON ,ON ,OFF,OFF,OFF,OFF];
   2    -> [ON ,ON ,OFF,ON ,ON ,OFF,ON ];
   3    -> [ON ,ON ,ON ,ON ,OFF,OFF,ON ];
   4    -> [OFF,ON ,ON ,OFF,OFF,ON ,ON ];
   5    -> [ON ,OFF,ON ,ON ,OFF,ON ,ON ];
   6    -> [ON ,OFF,ON ,ON ,ON ,ON ,ON ];
   7    -> [ON ,ON ,ON ,OFF,OFF,OFF,OFF];
   8    -> [ON ,ON ,ON ,ON ,ON ,ON ,ON ];
   9    -> [ON ,ON ,ON ,ON ,OFF,ON ,ON ];
  ^hA   -> [ON ,ON ,ON ,OFF,ON ,ON ,ON ];
  ^hb   -> [OFF,OFF,ON ,ON ,ON ,ON ,ON ];
  ^hC   -> [ON ,OFF,OFF,ON ,ON ,ON ,OFF];
  ^hd   -> [OFF,ON ,ON ,ON ,ON ,OFF,ON ];
  ^hE   -> [ON ,OFF,OFF,ON ,ON ,ON ,ON ];
  ^hF   -> [ON ,OFF,OFF,ON ,OFF,ON ,ON ];

end
```

Figure 3. ABEL Source Files for a Decoder

| FORM | PURPOSE |
|---|---|
| Declarations | Enter set names, pin assignments, etc. |
| Equations | Aid entry of Boolean equations |
| Truth Table | Aid entry of truth tables |
| State Diagrams | Aid entry of state diagram |
| Simulation | Set parameters, perform simulation |
| Reduction | Set parameters, perform reduction |
| Factoring | Set parameters, perform factoring |
| Partitioning | Define partition, display partitioning data |
| Schematic | Set parameters, perform schematic generation |
| PLD Map | Set parameters, create JEDEC file |

Each type of form has a predefined format and follows certain rules to make design entry easier, faster, and more accurate. For example, as an engineer enters an equation in the equations form, DASH-GATES checks each signal name against those entered on the declarations form. If a typing or assignment error occurs, an error message appears so the error can be corrected. For instance, entering an input on the output side of an equation would result in an error message, as would the use of an illegal operator or incorrect syntax.

The interactivity of the forms prevents design errors from accumulating in a design, only to be discovered later after much work. More than one form can be displayed on the screen at a time, and all forms are always "active"—that is, available to DASH-GATES for cross-checking of entries and collection of data for further processing.

The top of Figure 4 shows an equation form with three equations that describe the COUNT function of the counter/LED decoder design example. The first equation, "Count.d = (Count.q +1) & !Clear," describes the count-up operation that takes place only when Clear is low. Note that a ".d" or ".q" has been appended to the set name. This notation allows the engineer to explicitly state whether the reference is to the D input or Q output of a D flip-flop. It also provides better control of multiple feedback paths.

The second equation, "Count.clk = Clk", describes the clocking operation, assigning the clock input of the flip-flops to the Clk signal. The third equation, "Count.or = !OE1", is for the output enable.

## DASH-GATES Reduction

The lower half of figure 4 shows reduced equations. Note that reduction parameters can be entered in the four columns to the left of each equation to be reduced.

```
F1: HELP                          DASH-GATES                          Insert: off
>> edit              type:  reduction

Select reduction level:  None  Transform  Sum-of-products  Espresso
Reduce:  Done
      Form type:  equation      Form name:  count
 Count.d = (Count.q + 1) & !Clear

 Count.clk = Clk

 Count.oe  = !OE1



      Form type:  reduction     Form name:  Reduce
 Reduction Level
 ↓ Polarity
 ↓ ↓ Exclusive-OR
 ↓ ↓ ↓ Display
 e + n r  Q0.clk = Clk
 e - n r  !Q0.d = Clear & Q0.q
 e + n r  Q0.oe = !OE1
 e + n r  Q1.clk = Clk
 e - n r  !Q1.d = !Q1.q & !Q0.q & Q1.q & Q0.q & Clear
```

Figure 4. Split Screen Showing Original and Reduced Equations

```
F1: HELP                          DASH-GATES                          Insert: off
>> edit                  type:  simulation          name:  ledcount

Enter input or output value
                                                                F10 for new vector
      Form type:  simulation      Form name:  ledcount
      Clk Clear OE1 OE2 ! Count a  b  c  d  e  f  g
     ------------------------------------------------
  1   ~c  1     0   0     _0    _1 _1 _1 _1 _1 _1 _0
  2   ~c  0     0   0     _1    _0 _1 _1 _0 _0 _0 _0
  3   ~c  0     0   0     _2    _1 _1 _0 _1 _1 _0 _1
  4   ~c  0     0   0     _3    _1 _1 _1 _1 _0 _0 _1
  5   ~c  0     0   0     _4    _0 _1 _1 _0 _0 _1 _1
  6   ~c  0     0   0     _5    _1 _0 _1 _1 _0 _1 _1
  7   ~c  0     0   0     _6    _1 _0 _1 _1 _1 _1 _1
  8   ~c  0     0   0     _7    _1 _1 _1 _0 _0 _0 _0
  9   ~c  0     0   0     _8    _1 _1 _1 _1 _1 _1 _1
# 10  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 11  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 12  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 13  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 14  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 15  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 16  ~c  0     0   0     ____  __ __ __ __ __ __ __
# 17  ~c  0     0   0     ____  __ __ __ __ __ __ __
```

Figure 5. DASH-GATES Simulation in Progress

## DASH-GATES Simulation

A simulation form is shown in Figure 5 for the LED decoder portion of the design. Unlike ABEL, DASH-GATES will fill in the output section of the form automatically. Once the input values are entered, the simulator goes to the design description forms, applies the inputs to the design, obtains output values, and inserts them into the simulation form. Notice that in Figure 5 roughly half the simulation is complete, so half the values are filled in. The engineer checks the values to make sure they are correct and can then make them "permanent," so they may be used for checking future iterations of the design. With ABEL, all values must be entered manually.

Simulation takes place interactively and can be set to stop at the first error. An error message is displayed on the screen so corrections to the design can be made. In fact, because DASH-GATES can display more than one screen at a time, the engineer can simply call up the truth table for the decoder and make the appropriate change without leaving the simulation screen.

## DASH-GATES Factoring

PLDs differ in the number of inputs to their AND and OR gates, the number of product terms, the existence of feedback paths and internal registers, the number of inputs and outputs, and many other items of interest to the engineer. DASH-GATES' factoring algorithm optimizes the design equations for the gate counts of the target device, creating intermediate equations and multiple levels of logic to do so.

Figure 6 shows the counter equations before factoring; note that Q3.d requires 5 product terms. Figure 7 shows the factored equations for the counter outputs. During factoring, one intermediate equation, "cnt@0", was produced to reduce the number of product terms from 5 to 4. If an internal signal or extra input is available, this intermediate equation can be used. Note also that the intermediate equation introduced one more stage or level into the design (indicated by the [3] next to the equation). In timing-critical designs, such a tradeoff may not work; in other designs, saving one product term may mean cost and/or power savings by allowing the use of a smaller PLD.

```
F1: HELP                          DASH-GATES                          Insert: off
>> edit                  type:  factor

Select factor target:  Gate_array  Ifl  None Pal

     Form type:  factor     Form name:  Factor
                            Factored Output
 Group: cnt          AND Min: 2   AND Max: 20  OR Min: 3   OR Max: 4
                     Stage Count: 99  Factor Target: n  Polarity: +

Stages  Factored Equations:
    !Q0.d = Clear # Q0.q
    !Q1.d = !Q1.q & !Q0.q # Q1.q & Q0.q # Clear
    !Q2.d =
        Q2.q & Q1.q & Q0.q
      # !Q2.q & !Q0.q
      # !Q2.q & !Q1.q
      # Clear
    !Q3.d =
        Q3.q & Q2.q & Q1.q & Q0.q
      # !Q3.q & !Q0.q
      # !Q3.q & !Q1.q
      # !Q3.q & !Q2.q
      # Clear
```

Figure 6. Screen Showing Original Equations

```
F1: HELP                          DASH-GATES                          Insert: off
>> _______

Select factor target:  Gate_array  Ifl  None Pal

     Form type:  factor     Form name:  Factor
 Group: cnt          AND Min: 2   AND Max: 20  OR Min: 3   OR Max: 4
                     Stage Count: 99  Factor Target: p  Polarity: +

Stages  Factored Equations:
[1]     !Q0.d = Clear # Q0.q
[2]     !Q1.d = !Q1.q & !Q0.q # Q1.q & Q0.q # Clear
[2]     !Q2.d =
          Q2.q & Q1.q & Q0.q
        # !Q2.q & !Q0.q
        # !Q2.q & !Q1.q
        # Clear
[3]     !Q3.d = cnt@0 # Clear

Stages  Intermediate Equations:
[2]     cnt@0 =
          Q3.q & Q2.q & Q1.q & Q0.q
        # !Q3.q & !Q0.q
        # !Q3.q & !Q1.q
        # !Q3.q & !Q2.q
```

Figure 7. Screen Showing Factored Equations

## DASH-GATES Partitioning

Manually partitioning a PLD design into more than one PLD can be a difficult task. The engineer must determine which inputs and outputs are common to equations to decide what makes a sensible partition. DASH-GATES' partitioning form provides assistance in this task. Figure 8 shows a partitioning form for the full LEDCOUNT design.

The engineer enters design outputs that he thinks are candidates for partitioning. DASH-GATES references the various declaration and design description forms and fills in the required inputs, the required number of combinatorial and registered outputs, and the total number of pins used by the partition. In this case, it is obvious that there is a natural partition between the count function that produces the Q0 through Q3 outputs and the decoder function.

```
F1: HELP                          DASH-GATES                          Insert: off
>> edit              type:  partition          name:  ledcount

Enter output name (as in declaration)
                                                          F10 for new recor
     Form type:  partition     Form name:  ledcount
Partition: ledcount
Inputs: 4    Comb Outputs: 7 (0 w/feedback)   Reg Outputs: 4 (4 w/feedback)
Total Pins Required: 15

Output Name:                Output Type:   Inputs Required:
 Q3                               Bidir   Clear,Clk,OE1,Q0,Q1,Q2,Q3
 Q2                               Bidir   Clear,Clk,OE1,Q0,Q1,Q2
 Q1                               Bidir   Clear,Clk,OE1,Q0,Q1
 Q0                               Bidir   Clear,Clk,OE1,Q0
 a                                Output  OE2,Q0,Q1,Q2,Q3
 b                                Output  OE2,Q0,Q1,Q2,Q3
 c                                Output  OE2,Q0,Q1,Q2,Q3
 d                                Output  OE2,Q0,Q1,Q2,Q3
 e                                Output  OE2,Q0,Q1,Q2,Q3
 f                                Output  OE2,Q0,Q1,Q2,Q3
 g                                Output  OE2,Q0,Q1,Q2,Q3
```

Figure 8. Partitioning Form Used to Partition Large Designs

```
F1: HELP                        DASH-GATES                          Insert: off
>> edit              type:  pld-map            name:  count

Enter PLD device type

      Form type:  pld-map     Form name:  count
Partition: count
Target Device Type: P16R8       Output file format: jedec
Output File Name: count.jed
File Title Line: 4 bit binary counter  FutureNet a Data I/O Company
Checksum Format: full            Fast Flag: no
Simulation Form(s): ledcount

                Special Fuse Number:        Value:
                                    ------

Pin Name:                       Type:  P-terms:  Pin Number:  Active Level:
Clear                            Input    ___        2            high
Clk                              Input    ___        1            high
OE1                              Input    ___       11            high
Q0                               Bidir    2         14            high
Q1                               Bidir    3         15            high
Q2                               Bidir    4         16            high
Q3                               Bidir    5         17            high
```

**Figure 9. Device and Pin Assignment Form Used When a GATES Design is Programmed Into a PLD**

### Pin and Device Assignment

To this point, the DASH-GATES design description has been completely "technology independent." In other words, the function of the design has been described without regard to the type of device used to implement it. This design could be part of a larger gate array or a complete program for a PLD. In this case, we have a partitioned design for two PLDs and must assign signals to the PLD pins. Figure 9 shows a device and pin assignment form.

## Two Powerful Tools: ABEL and DASH-GATES

As the above discussion shows, both ABEL and DASH-GATES are powerful PLD design tools that address the needs of the PLD design process. Both provide natural design entry methods, automated logic reduction and simulation, and full design documentation.

The differences between ABEL and DASH-GATES arise mainly in the way a design is entered and processed. ABEL is a character-oriented, batch process; DASH-GATES is a graphical, interactive tool. DASH-GATES also features advanced logic reduction, factoring and partitioning and schematic generation.

The choice between ABEL or DASH-GATES is a choice determined by a variety of factors, among them budget, need, and hardware availability. ABEL will serve many engineers to the full extent they need; others will decide the advanced processing of DASH-GATES warrants the additional investment. The choice of tools is simply a choice between two very high levels.

# CUPL

## Complier-Based Software and PLDs Improve Logic Design

**Part 1**

*Programmable logic devices allow you to complete a design faster than you can using SSI devices or custom ICs, and PLD implementations take up less space than do SSI-based circuits. Moreover, easy-to-use compiler-based languages that don't require you to understand PLD architectures make PLDs increasingly attractive for logic designs.*

Bob Osann, *Assisted Technology*

Circuits that incorporate programmable logic devices (PLDs) take up less board space than do SSI-based implementations and require less design time than do custom-IC or SSI-based versions. But until recently, the PLDs' unusual architecture and lack of software support made designers hesitant to use the devices, despite the advantages they offer. Compiler-based software, however, is simplifying PLD use; this high-level software makes it unnecessary for you to be concerned with the PLDs' internal details when implementing logic functions with the devices.

This first article in this 3-part series, which is aimed at first-time PLD users, discusses basic PLD architecture and shows you how to replace two simple logic designs with PLDs using a compiler-based PLD design language. Part 2 will show you how to replace more complicated combinatorial and registered-TTL designs with PLDs. Part 3 will introduce the state-machine concept and show you how to implement a logic design directly, without ever developing a gate-level description of the system.

Although the PLD approach lets you go from logic function to PLD circuit without conceiving a gate-level description, when designers decide to use PLDs, they usually have either completed TTL designs that they want to shrink or else gate-level descriptions of circuits they don't want to implement in discrete logic. Therefore, the first two articles in this series target converting existing designs.

### Why use a PLD?

For one-of-a-kind designs, prototypes, or small production runs, designers have traditionally taken the discrete approach. Discrete designs are easy to modify and inexpensive to manufacture in small quantities, and you can complete them more quickly than you can complete custom or semicustom designs. For production runs over 500, designers have typically chosen the semicustom and custom routes and sacrificed short design cycles and ease of modification to reduce manufacturing costs.

PLDs bridge the gap between bulky discrete designs and long custom-IC design cycles. On the one hand, PLD designs are easier to modify than SSI-based ones

EDN January 10, 1985

*A PLD approach allows designers to go from a logic function to a PLD-based circuit without conceiving a gate-level description.*

and use much less space. Moreover, depending on the application, they can cost less than SSI-based implementations for even small production runs. And on the other hand, although custom ICs can prove more economical than PLDs for large production runs, PLD design cycles are much shorter. So, if you need to get a small, inexpensive design to market quickly and can't wait for a completed custom design, PLDs can provide you with a quick stand-in until your custom design is completed.

In general, the PLD architecture contains a fixed logic array made of AND gates—whose outputs feed OR gates—and a programming matrix. The programming matrix is made up of fuses that you blow with a programming device. By blowing the appropriate fuses, you can achieve any AND/OR product or combination. **Fig 1** shows the PAL-type and FPLA-type architectures. The total number of terms that you can generate is limited only by the size of the matrix.

Because you can represent any logical function as the logical sum of product terms, you can realize any logical function using a PLD. A product term consists of any combination of input variables or their complements ANDed together. A logical sum is any combination of

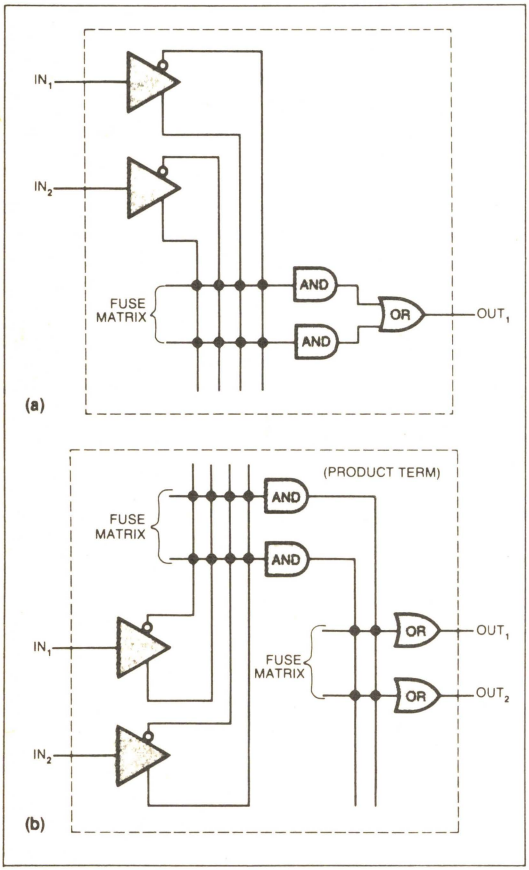



***Fig 1—Typical PLDs use one of two general architectures** to permit implementation of a wide range of logic functions. PAL-type devices **(a)** prove easier to use, but FPLAs **(b)** provide more flexibility by allowing two levels of programmability.*

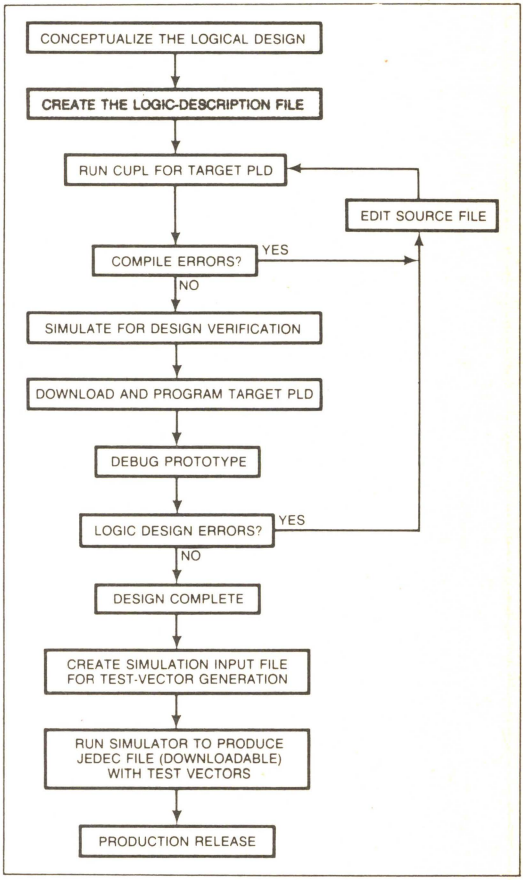



***Fig 2—PLDs greatly simplify logic design.** After you complete the logic-description file, the PLD software automatically compiles the data for downloading to a programming device.*

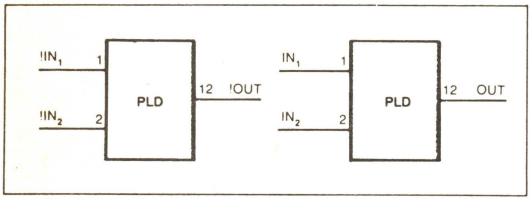



***Fig 3—When using CUPL,*** *you can always write your logic equations in positive logic, regardless of the actual polarity of the signals entering the device. For example, the two cases illustrated above both yield the same logic equation: OUT=IN1&IN2.*

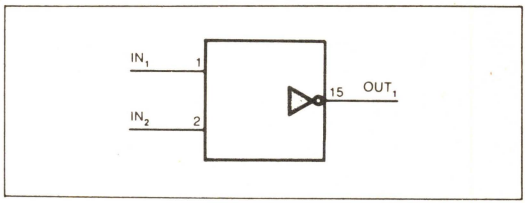



***Fig 4—Some PLD devices use an inverting output buffer.*** *As a result, to accommodate applications that demand an active-high output signal, the compiler often must generate extra product terms that might make the design too big for the target PLD.*

product terms ORed together. Using De Morgan's theorems,

$$(\overline{AB}) = \overline{A} + \overline{B}, \text{ and}$$
$$(\overline{A + B}) = \overline{A}\ \overline{B}.$$

Then, using the distributive property,

$$A\,(B + C) = AB + AC, \text{ and}$$
$$(A + B)(C + D) = AC + AD + BC + BD.$$

The PLD software determines the best form of the equation that will fit into a PLD, which uses a general architecture to permit implementation of a wide range of functions. The software should allow you to think in terms of logical functions rather than gates. The better the software, the more you can abstract from the details of discrete design and attend to system concerns.

Once you've decided to use a PLD approach, you'll need to choose the software development support for that device. You can use two basic types of software: assembler-based software and compiler-based software **(Ref 1).** Assembler-based software is supplied by the PLD manufacturer; it typically supports only that manufacturer's devices. If you buy PLDs in large quantity, you can usually get the software for well under $100. An alternative to assembler-based software is the compiler-based software sold by Data I/O and Assisted Technology. Compiler-based software supports almost all PLD devices and programmers; typical prices range from $750 for a version that runs on CP/M-based systems to $2695 for a version that runs on VAX/VMS systems.

Although compiler-based software is more expensive, it will make your PLD design task easier. Capabilities such as symbolic signal representation and macro substitution make it easier for you to formulate and enter your logic equations. These improvements allow you to formulate your design at a higher conceptual level; that is, you can think in terms of systems instead of individual circuits.

**Fig 2** illustrates the PLD design process using Assisted Technology's CUPL language. (The Abel language, developed by Data I/O, could also be used to demonstrate the techniques involved.)

### The CUPL syntax

Before you can design with CUPL, you have to learn the syntax. CUPL's operators, which were chosen largely from the C programming language, are as follows:

&=logical AND
#=logical OR
$=logical exclusive-OR
!=logical negation.

You can place comments anywhere within a CUPL logic specification by using the symbol /* for "start comment" and the symbol */ for "end comment." You can also nest parentheses to any level, as in this example: OUT=!((A&B)&(C#(D&E))).

To facilitate clear documentation, CUPL allows you to use symbolic names of arbitrary length (the first 31 characters must be unique). Symbolic names can represent pin variable names, internal device nodes, intermediate variables, bit-field representations, and symbolic constants. To further improve clarity, you can use the underscore character—

RAM_PARITY_INT_EN.

When you're converting an existing design, CUPL allows you to give symbolic names to internal nodes within your design. For example, for flip-flops connected to the pin PIN_VAR, you would name the node as follows:

- D-type flip-flop—PIN_VAR.D=Expression

*The PLD architecture contains a fixed logic array made of a programming matrix and AND gates whose outputs feed OR gates.*

- JK-type flip-flop—PIN_VAR.J=Expression, PIN_VAR.K=Expression
- RS-type flip-flop—PIN_VAR.R=Expression, PIN_VAR.S=Expression.

For 3-state-device enable signals connected to a pin, you would write:

- PIN_VAR.OE=Expression
- [PIN_VAR LIST].OE=Expression,

as in [DATA7..0].OE=Expression. If you're leaving the 3-state device enabled, you don't have to write an equation for it.

### Handling signal polarities

One issue that often confuses first-time PLD users is the representation of signal polarities. In CUPL, you can always write equations in positive logic, regardless of the polarity of the signals entering the device. Because all signals entering the PLD are buffered, you have access to both the true and complement versions of the input signal for your logic equations. **Fig 3** illustrates two simple cases. For each case—if you were using the PLD as an AND gate—you would write the same logic equation: $OUT=IN_1\&IN_2$.

The specification of signal polarities is complicated by the inverting-output architecture of, for instance, 20-pin PAL devices (**Fig 4**). If you need an active-low output polarity, this doesn't create a problem. In this case, the compiler has to implement only one P (product) term. However, if you need an active-high output signal, the compiler must apply De Morgan's theorem,

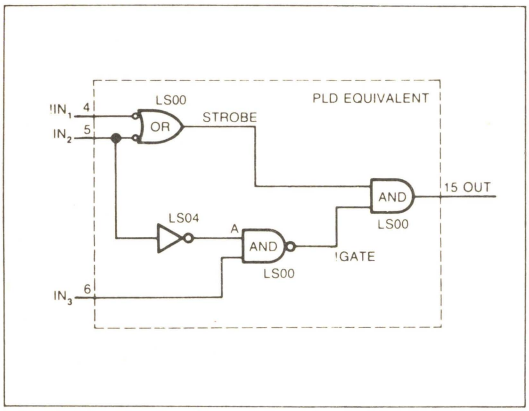



***Fig 5—With CUPL, you can often replace a TTL design without understanding its function.*** *You just name the pins and nodes, combine them according to gate relationships in the circuit, and the software does the rest.*

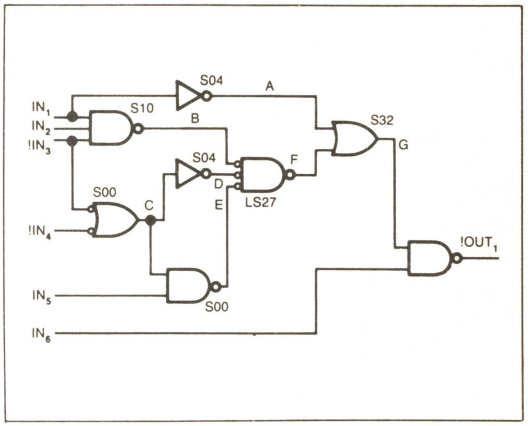



***Fig 6—Reduced propagation delays*** *are one of the benefits of using PLDs. A PLD implementation of the circuit shown here has, on the average, half the propagation delay of the discrete implementation.*

and $!OUT_1=!(IN_1\&IN_2)$ becomes $!IN_1\#!IN_2$. Note that this equation contains two product terms. The additional space the compiler requires reduces the probability that the compiler will be able to fit the logic function into the target PLD.

CUPL can eliminate this problem for PLD devices that have programmable output polarities. CUPL automatically chooses the output polarity that will result in the fewest number of P terms.

### Reduce keystrokes

One of CUPL's (and Abel's) major advantages is macro substitution, the ability to use a single variable name to represent a complex logical equation. For example, if you define "INT_VAR" as "A&B#C," the compiler will insert A&B#C every time it encounters INT_VAR.

Because macro substitution lets you use fewer keystrokes to write equations, it saves time and reduces the probability that you'll make input errors. By using macro substitution, you can write your logic specification in a hierarchical fashion, breaking complex equations into more manageable and readable pieces.

### The logic description

The heart of CUPL is the logic-description file (LDF), which contains your logic equations, pin declarations, intermediate variables, and documentation describing the device's function. You must complete the LDF to prepare your logic equations for downloading to

a programming device. **Table 1** shows the format for a CUPL LDF that was written for a memory decoder.

The following example shows you how to complete the logic equation, pin declaration, and intermediate variable portions of an LDF for the design in **Fig 5.** First, you write the pin declarations using the same names and signal polarities that appear on your schematic. Next, you name the output of each gate in the schematic. In the example, STROBE, A, and !GATE are the intermediate variables. Using the intermediate variable definitions, you then write an equation for the output:

$$\begin{aligned}
\text{PIN } 4 &= !IN_1 \\
\text{PIN } 5 &= IN_2 \\
\text{PIN } 6 &= IN_3 \\
\text{PIN } 15 &= OUT \\
A &= !IN_2 \\
STROBE &= !(!IN_1)\#!IN_2;\ /*!(!IN_1) = IN_1*1 \\
!GATE &= !(A\&IN_3) \\
OUT &= STROBE\&!GATE.
\end{aligned}$$

The following expressions show this strategy applied to the more complicated design in **Fig 6**:

$$\begin{aligned}
A &= !IN_1 \\
B &= !(IN_1\&IN_2\&!IN_3) \\
C &= !(!IN_3)\#!(!IN_4) \\
D &= !C \\
E &= !(C\&IN_5) \\
F &= !B\&!D\&!E \\
G &= A\#F \\
!OUT &= (G\&IN_6).
\end{aligned}$$

The design in **Fig 6** illustrates another advantage of using PLDs instead of discrete logic. The propagation delay in the PLD implementation is often less than that in the discrete design. The discrete design for this circuit requires at least three TTL packages and has five levels of delay. The total delay time is 50 nsec (five levels times 10 nsec/level) for LS packages and 26 nsec (4×4 nsec+10 nsec) for a combination of LS and Schottky TTL packages. In an equivalent PLD circuit, the maximum delay is 25 nsec; typical delay is only 15 nsec.

### Registered PLDs

Some of the more complicated types of PLDs use flip-flops in their output stages to store information. Most of these PLDs provide integral feedback paths. The simplest registered PLDs contain D-type flip-flops, which transfer the signal at their D input to their Q output after one clock pulse (more specifically, after the application of a positive-going leading edge). The equations for the flip-flop in **Fig 7** are

```
OUTPUT.D=G&INPUT /*UPDATE WITH INPUT*/
        #!G&OUTPUT; /*MAINTAIN CURRENT OUTPUT*/
                    /*VIA INTERNAL FEEDBACK DATA*/.
```

For simple registered designs, you can often model

## TABLE 1—SOURCE SPECIFICATION FILE FORMAT

| FUNCTION | | DESCRIPTION |
|---|---|---|
| PART NO | 900 16487 | HEADER INFORMATION: IDENTIFIES THE PARTICULAR LOGIC SOURCE FILE |
| NAME | MEMDEC | |
| DATE | 07/18/84 | |
| REV | 03 | |
| DESIGNER | OSANN | |
| COMPANY | ATI | |
| ASSEMBLY | PC-RAM | |
| LOCATION | 417 | |
| THIS DEVICE DECODES ADDRESSES FOR THE DYNAMIC RAM AND PROVIDES THE RAS STROBES AS WELL AS A SIGNAL THAT INITIATES CAS. | | TITLE BLOCK: DESCRIBES IN PLAIN TERMS WHAT THIS DEVICE DOES. |
| ALLOWABLE TARGET DEVICE TYPES: PAL 16L8, 825153, EP300. | | DEVICE MENU: LISTS ALL TARGET DEVICE TYPES THAT MAY BE USED. |
| INPUTS: | | PIN DECLARATIONS: |
| PIN [1..6] = [A 19 . . 14] | | CPU ADDRESS BUS |
| PIN [7, 8] = ! [MEMW, MEMR] | | MEMORY DATA STROBES |
| PIN 9 = ! REF_ADR_EN | | INDICATES REFRESH CYCLE IN PROGRESS |
| PIN 11 = ! REF_RAS | | STROBE FOR RAS-ONLY REFRESH |
| PIN 13 = ALT_LOC | | PLACE MEMORY IN ALTERNATE RANGE |
| OUTPUTS: | | |
| PIN [19 . . 16] = ! [RAS 3 . . 0] | | RAM ROW ADDRESS STROBES |
| PIN 14 = ! CAS_INIT | | ENABLE CAS STROBES |
| DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS: | | WRITE EQUATIONS FOR BIT-FIELD DECLARATIONS AND INTERMEDIATE VARIABLES WHICH WILL BE SUBSTITUTED LATER USING MACRO-SUBSTITUTION. |
| FIELD MEMADR = [A19 . . A14] | | MEMORY ADDRESS |
| MEM REQ = MEMW # MEMR | | MEMORY REQUEST |
| LOGIC EQUATIONS: | | WRITE EQUATIONS FOR OUTPUTS IN TERMS OF INPUTS AND FEEDBACK AS IN: OUTPUT = INPUT 1 & FEEDBACK 1 # INPUT 2 & FEEDBACK 2 # INPUTS N & FEEDBACK N |

| FUNCTION | DESCRIPTION |
|---|---|
| RAS 3 = MEMREQ & ! REF_ADR_EN & | |
| (! ALT_LOC & MEMADR: [0C000 . . 0FFFF] | PRIMARY RANGE |
| # ALT_LOC & MEMADR: [FC000 . . FFFFF) | ALTERNATE RANGE |
| # REF_ADR_EN & REF_RAS | REFRESH CYCLE |
| RAS 2 = MEMREQ & ! REF_ADR_EN & | |
| (! ALT_LOC & MEMADR: [08000 . . 0BFFF] | PRIMARY RANGE |
| # ALT_LOC & MEMADR: [F8000 . . FBFFF) | ALTERNATE RANGE |
| # REF_ADR_EN & REF_RAS | REFRESH CYCLE |
| RAS 1 = MEMREQ & ! REF_ADR_EN & | |
| (! ALT_LOC & MEMADR: [04000 . . 07FFF] | PRIMARY RANGE |
| # ALT_LOC & MEMADR: [F4000 . . F7FFF) | ALTERNATE RANGE |
| # REF_ADR_EN & REF_RAS | REFRESH CYCLE |
| RAS 0 = MEMREQ & ! REF_ADR_EN & | |
| (! ALT_LOC & MEMADR: [00000 . . 03FFF] | PRIMARY RANGE |
| # ALT_LOC & MEMADR: [F0000 . . F3FFF) | ALTERNATE RANGE |
| # REF_ADR_EN & REF_RAS | REFRESH CYCLE |
| CAS_INIT = MEMREQ & ! REF_ADR_EN & | |
| (! ALT_LOC & MEMADR: [00000 . . 0FFFF] | PRIMARY RANGE |
| # ALT_LOC & MEMADR: [F0000 . . FFFFF) | ALTERNATE RANGE |

*Compiler-based software for PLD design includes such features as symbolic signal representation and macro substitution.*

the circuit with a timing diagram. Using the timing diagram, you can write your logic equations easily. In the **Fig 8** timing diagram for a D-type flip-flop, $INPUT_2$ initiates the input pulse, and $INPUT_1$ terminates the output pulse. The pin declarations are

```
PIN 3=!INPUT1
PIN 6=!INPUT2
PIN 1=CLOCK
PIN 14=OUTPUT,
```

and the corresponding logic equations are

```
OUTPUT.D=!OUTPUT&INPUT 2 /*SET FF*/
        # OUTPUT&!INPUT 1; /* KEEP FF SET*/
                           /* UNTIL INPUT 1*/
                           /*GOES ACTIVE*/.
```

These equations demonstrate one method for using the smallest possible number of product terms to keep a D flip-flop set for several clock cycles. Here, the flip-flop's output is fed back until some condition is met that again enables the flip-flop.

If the registered PLD contained JK flip-flops, the expressions would be

```
OUTPUT.J=INPUT2; /* SET FF*/
OUTPUT.K=INPUT1; /* RESET FF*/.
```

To handle more complicated sequential designs, you can model your circuit as a multiple-flip-flop system that uses a common clock. (Virtually all currently available registered PLDs use common clocks for their flip-flops.) For example, to convert TTL designs that use cascaded flip-flops (in which the outputs of some flip-flops are used to clock other flip-flops), you must find the originating clock in the circuit, which is usually

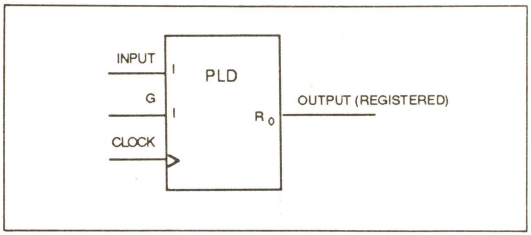



***Fig 7—Some PLDs use registered outputs*** *to introduce storage elements into their architecture.*

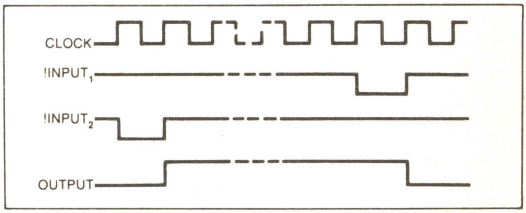



***Fig 8—Converting logic designs to PLDs is easy*** *once you've completed a timing diagram for your circuit. This one represents operation of a D-type flip-flop.*

## TABLE 2—CUPL OPTION FLAGS

| | |
|---|---|
| A | PRODUCE YOUR_FILE_NAME.ABS FOR LATER USE BY CSIM. |
| L | PRODUCE YOUR_FILE_NAME.LST WITH LINE NUMBERS AND ERROR MESSAGES. |
| I | PRODUCE YOUR_FILE_NAME.HL DOWN-LOADABLE HL FORMAT FILE FOR IFL. |
| H | PRODUCE YOUR_FILE_NAME.HEX MMI PAL ASCII-HEX FORMAT FILE. |
| F | PRODUCE YOUR_FILE_NAME.DOC WITH FUSE MAP FILE. |
| X | PRODUCE YOUR_FILE_NAME.DOC WITH FULLY EXPANDED EQUATIONS. |
| G | PROGRAM SECURITY FUSE. |
| R | DISABLE GLOBAL PRODUCT-TERM MERGING. (FPLA DEVICES). |
| M0 | PERFORM NO LOGIC MINIMIZATION. |
| M1 | PERFORM LOCAL LOGIC MINIMIZATION. |
| M2 | PERFORM LOGIC MINIMIZATION UNTIL EQUATIONS FIT IN TARGET DEVICE. |
| M3 | PERFORM FULL LOGIC MINIMIZATION. |
| D | DEACTIVATE UNUSED OR-TERMS. (INCREASES SPEED IN FPLAs). |
| U | SET ALTERNATE SEARCH PATH FOR PLD DEVICE DATABASE. |
| J | PRODUCE YOUR_FILE_NAME..IED, THE JEDEC FORMAT DOWNLOADABLE FILE |
| S | AUTOMATICALLY RUN CSIM AFTER RUNNING CUPL |

*Compiler-supported symbolic names can represent pin variable names, internal device nodes, intermediate variables, bit-field representations, and symbolic constants.*

the highest-frequency source in the circuit. In most cases, the timing skew from one flip-flop output to the next is tolerable.

The TTL circuit in **Fig 9** contains an LS161 counter whose output is decoded in an LS138. The decoded output sets and resets flip-flops at various points in the timing cycle. The timing diagram in **Fig 10** is based on the assumption that the clock rate is sufficiently high that the propagation delays from SYSCLK to $OUT_1$ and $OUT_2$ are not significant. If you were to implement this design in a PLD, the pinout would look like the one shown in **Fig 11.** Outputs $Q_0$ and $Q_1$ were added to make all eight time slots in the circuit's cycle a unique combination of the four outputs. Adding $Q_0$ and $Q_1$ results in a timing sequence like the one in **Fig 12.**

You can now write the logic equations by noting, for each output, each place in the timing cycle where the output reads high (the flip-flop is set). For example, $OUT_1$ is set during time slots 2, 3, and 4. (The equation for the D input should include representations of time slots 1, 2, and 3; these time slots occur immediately before the flip-flop is set.) For time slots 1 through 3, you can now write

```
OUT1.D=!OUT1&!OUT2&Q0&!Q1 /*TIME SLOT 1*/
      #OUT1&!OUT2&!Q0&!Q1 /*TIME SLOT 2*/
      #OUT1&!OUT2&Q0&!Q1 /*TIME SLOT 3*/.
```

Writing these equations is easier if you first define each time slot in terms of the register outputs that are fed

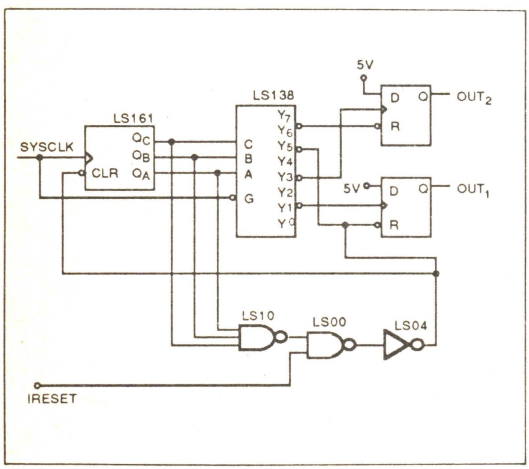



***Fig 9—When converting complex sequential designs to PLDs,*** *you can model your circuit as a group of flip-flops driven by a common clock.*

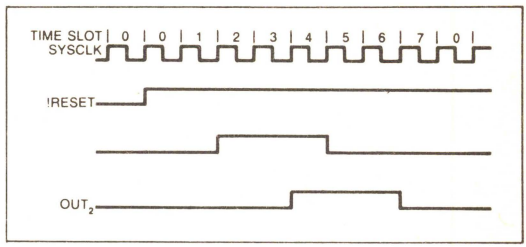



***Fig 10—This timing diagram*** *is based on the assumption that the **Fig 9** circuit uses a clock rate that is not significantly affected by propagation delays from SYSCLK to $OUT_1$ and $OUT_2$.*

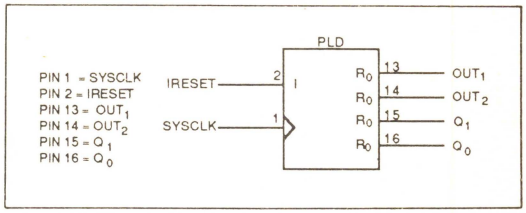



***Fig 11—Adding outputs $Q_0$ and $Q_1$*** *of this PLD implementation of the **Fig 9** circuit makes each of the eight intervals in the **Fig 12** timing cycle a unique combination of the circuit's four outputs.*

back into the programmable array:

```
TS0=!OUT1&!OUT2&!Q0&!Q1; /*TIME SLOT 0*/
TS1=!OUT1&!OUT2&Q0&!Q1; /*TIME SLOT 1*/
TS2=OUT1&!OUT2&!Q0&!Q1; /*TIME SLOT 2*/
TS3=OUT1&!OUT2&Q0&!Q1; /*TIME SLOT 3*/
TS4=OUT1&OUT2&!Q0&!Q1; /*TIME SLOT 4*/
TS5=!OUT1&OUT2&!Q0&!Q1; /*TIME SLOT 5*/
TS6=!OUT1&OUT2&Q0&!Q1; /*TIME SLOT 6*/
```

You can now write the equations for the four registered outputs in terms of $TS_0$ through $TS_6$ ($TS_7$ is not needed); CUPL performs the following substitutions:

```
OUT1.D=TS1#TS2#TS3
OUT2.D=TS3#TS4#TS5
  Q0.D=TS0#TS2#TS5
  Q1.D=TS6.
```

### Running CUPL

Once you've completed the LDF, you're ready to compile the LDF for downloading to the PLD programmer. To compile the file, you type an expression that follows this format:

CUPL [FLAGS] TARGET_DEVICE_CODE YOUR_FILE_NAME.

*PLDs with an inverting-output architecture complicate selection of signal polarities.*

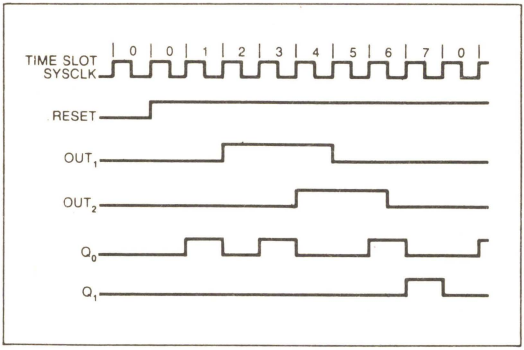



***Fig 12—Once you've rewritten the Fig 10 timing diagrams** to reflect the PLD configuration in **Fig 11**, you can write a set of logic equations for implementing the PLD design.*

For example, the sequence CUPL -J -A P16L8 RAM-CNTRL compiles the source file for a RAM controller that is targeted for a PAL16L8. The J and A symbols are chosen from a table of CUPL option flags (**Table 2**). In this case, the compiler produces a JEDEC file and an absolute-format file to be used later by CUPL's simulator, CSIM (**Ref 1**). The resulting compiled code is downloaded to the programmer, which then blows the appropriate fuses in the PLD.

The designs discussed thus far are simple but useful for describing the PLD design process. The next two articles will extend the discussion to more advanced designs, and finally, to the state-machine approach.

**EDN**

## Reference

Marrin, K, "Programmable logic devices gain software support," *EDN*, February 9, 1984, pg 67.

# CUPL

## Use PLDs to Shrink Complex, Discrete Logic Designs — Part 2

*As discrete combinatorial and sequential logic circuits become more complex, it becomes more difficult to convert them to PLD equivalents. With the help of compiler-based software, though, you'll be converting complicated logic designs in no time.*

Bob Osann, *Assisted Technology Inc*

Converting complicated discrete designs to their PLD (programmable logic device) equivalents can be an imposing task for the first-time PLD user or for the engineer who's been laboring with outmoded PLD software tools. New compiler-based software, however, makes it easy for you to implement even complex logic designs with PLDs.

This article, the second in a 3-part series on PLD design, introduces a few of the more advanced features of the compiler-based PLD design language CUPL and shows you how to use those features to convert complicated sequential and combinatorial SSI logic designs to PLD equivalent designs. Part 1 of the series demonstrated some elementary features of CUPL and showed you how to apply those features in a few simple designs. Part 3 will introduce CUPL's state-machine syntax and show you how to move directly from logic ideas to PLD implementations without developing a gate-level description of your system.

**CUPL lets you use a systems approach**

The CUPL high-level PLD support language enables you to develop your logic designs using a systems approach. This approach not only speeds the design process but facilitates the generation of logic descriptions that are easy to understand.

CUPL supports a systems approach with several advanced features, which give you a self-documenting syntax, allow you to use fewer keystrokes to develop your systems, and let you use symbolic names that correspond to whatever function you're trying to implement. CUPL also gives you a flexible format, which lets you describe several similar systems in less time than it would take to describe the systems using a more rigid format.

One of CUPL's advanced features is its bit-field capability, which allows you to use a single symbolic name to represent a group of bits (such as an address bus or state bit field). This feature saves you keystrokes when you're formulating your design equations and makes the resulting equations easier to read. Once you've defined a symbolic name, you can use that name to represent either a single hexadecimal value or a range of hexadecimal values. For example, in an address-decoding application, you could equate the symbolic name MEMADR with [ADR7, ADR6, ADR5,

*PLDs are effective replacements for both simple and complex combinatorial and sequential discrete logic designs.*

ADR4, ADR3, ADR2, ADR1, ADR0]. You could then substitute [ADR7 . . . 0] for [ADR7, ADR6, ADR5, ADR4, ADR3, ADR2, ADR1, ADR0]. The resulting equation, FIELD MEMADR=[ADR7 . . . ], assigns the name MEMADR to the address bus.

### CUPL speeds bit-field comparisons

Another CUPL feature is its ":" operator, which can perform bit-field comparisons and operations quickly and efficiently. This feature is particularly useful for describing such features as an address decoder. When the compiler is performing a bit-field comparison, the operator ":" compares a bit field with either a hexadecimal or an octal constant value or a hexadecimal or octal list of constant values (hexadecimal is the default value). When you're describing an address decoder, for example, the statement MEMADR: [A000 . . . EFFF] is true if the address MEMADR falls in the hexadecimal range A000 to EFFF (inclusively). Note that hexadecimal constant values must contain the proper number of nibbles to include the most significant bit of the bit field. In the above expression, the most significant bit of the E in EFFF corresponds to A15 in MEMADR.

You can also use the ":" operator for bit-field operations, as in the following equation:

```
IOADR: & REPLACES A7&A6&A5&A4&A3&A2&A1&A0
IOADR: # REPLACES A7#A6#A5#A4#A3#A2#A1#A0.
```

Another timesaving CUPL feature is the preprocessor, which lets you write general-purpose logic descriptions that you can tailor to suit more than one application. For example, you might write a general-purpose decoder that you could adapt to 8-, 16-, or 32-bit applications by changing a few symbolic names and ranges.

The CUPL preprocessor is a program that operates on the CUPL source file before it's compiled. The preprocessor's string-substitution function, for example, can replace one symbolic name with another until some condition is met. When it encounters the statement $DEFINE ARG1 ARG2, for instance, the preprocessor replaces ARG1 with ARG2 until it encounters the statement $UNDEF ARG1. You could use the arguments in this example to represent different versions of your decoder. You could make ARG1 represent, say, the 8-bit decoder, and you could make ARG2 represent the 16-bit decoder.

The preprocessor also allows you to delay inclusion of a file until compile time. Again, this feature lets you generalize your functions. For example, you could write several files that represent several specific cases of a general application. To implement different functions, you'd just include different file names. In the statement $INCLUDE FILENAME, the referenced file becomes part of the LDF (logic description file) only at compile time.

Conditional control structures extend even further the ability to create generalized files. They allow you to

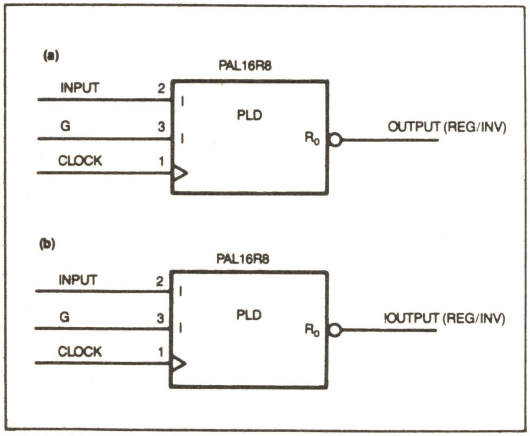



***Fig 1—These two PLDs show two possible output configurations for a PLD with a fixed inverting output buffer. PLDs with programmable output polarity eliminate the confusion that fixed output devices cause.***

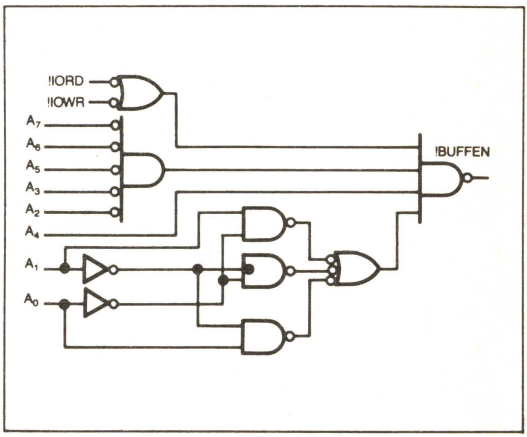



***Fig 2—Address decoders are typical targets for first-time PLD users.*** *A simple application like this address decoder shows how you can benefit from software features like macro substitution, range functions, and list notation.*

## TABLE 1 — MEMDEC LOGIC DESCRIPTION FILE

```
PARTNO      2600A00004 ;
NAME        MEMDEC ;
DATE        02/14/84 ;
REV         02 ;
DESIGNER    OSANN ;
COMPANY     ASSISTED TECHNOLOGY ;
ASSEMBLY    PC-RAM ;
LOCATION    U76 ;

/*****************************************************************************/
/*  THIS DEVICE DECODES ALL MEMORY ACCESSES FOR BOTH PRIMARY AND             */
/*  ALTERNATE LOCATIONS. IT GENERATES THE RAS SIGNALS FOR THE FOUR           */
/*  BANKS OF 16K DYNAMIC RAMS AS WELL AS THE SIGNAL THAT INITIATES           */
/*  THE CAS SIGNALS.                                                         */
/*****************************************************************************/
/** ALLOWABLE TARGET DEVICE TYPES: PAL16L8, 82S153, PAL16P8                 **/

/** INPUTS **/
PIN [1..6]          = [A19..14]             ; /* CPU ADDRESS BUS */
PIN [7,8]           = ![MEMW,MEMR]          ; /* MEMORY DATA STROBES */
PIN 9               = !REF_ADR_EN           ; /* INDICATES REFRESH CYCLE IN PROGRESS */
PIN 11              = !REF_RAS              ; /* STROBE FOR RAS-ONLY REFRESH */
PIN 13              = ALT_LOC               ; /* PLACE MEMORY IN ALTERNATE RANGE */

/** OUTPUTS **/
PIN [19..16]        = ![RAS3..0]            ; /* RAM ROW ADDRESS STROBES */
PIN 14              = !CAS_INIT             ; /* ENABLE CAS STROBES */

/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/

FIELD MEMADR        = [A19..14]             ; /* MEMORY ADDRESS */

MEMREQ              = MEMW # MEMR           ; /* MEMORY REQUEST */

/** LOGIC EQUATIONS **/
RAS3        =   MEMREQ & !REF_ADR_EN &
                (!ALT_LOC & MEMADR:[0C000..0FFFF]       /* PRIMARY RANGE */
                # ALT_LOC & MEMADR:[FC000..FFFFF])      /* ALTERNATE RANGE */
                # REF_ADR_EN & REF_RAS ;                /* REFRESH CYCLE */

RAS2        =   MEMREQ & !REF_ADR_EN &
                (!ALT_LOC & MEMADR:[08000..0BFFF]       /* PRIMARY RANGE */
                # ALT_LOC & MEMADR:[F8000..FBFFF])      /* ALTERNATE RANGE */
                # REF_ADR_EN & REF_RAS ;                /* REFRESH CYCLE */

RAS1        =   MEMREQ & !REF_ADR_EN &
                (!ALT_LOC & MEMADR:[04000..07FFF]       /* PRIMARY RANGE */
                # ALT_LOC & MEMADR:[F4000..F7FFF])      /* ALTERNATE RANGE */
                # REF_ADR_EN & REF_RAS ;                /* REFRESH CYCLE */

RAS0        =   MEMREQ & !REF_ADR_EN &
                (!ALT_LOC & MEMADR:[00000..03FFF]       /* PRIMARY RANGE */
                # ALT_LOC & MEMADR:[F0000..F3FFF])      /* ALTERNATE RANGE */
                # REF_ADR_EN & REF_RAS ;                /* REFRESH CYCLE */

CAS_INIT    =   MEMREQ & !REF_ADR_EN &
                (!ALT_LOC & MEMADR:[00000..0FFFF]       /* PRIMARY RANGE */
                # ALT_LOC & MEMADR:[F0000..FFFFF]);     /* ALTERNATE RANGE */
```

3

*By using symbolic names to represent bit fields such as address buses, you can not only save keystrokes, but you can make your designs virtually self documenting.*

compile particular portions of your LDF when you've complied with certain conditions. When you use the format

```
$IFDEF ARG
            . . . STATEMENTS . . .
$ELSE
            . . . STATEMENTS . . .
$ENDIF,
```

the statements are compiled only if the argument ARG has been defined. When you use the format

```
$IFNDEF ARG
            . . . STATEMENTS . . .
$ELSE
            . . . STATEMENTS . . .
$ENDIF,
```

the statements are compiled only if the argument ARG has not been defined.

**Output programmability saves space**

One CUPL feature that can save you considerable space in your design is the language's ability to support a PLD with programmable output polarity. For PLDs with this feature, the CUPL compiler chooses whichever output polarity results in logic equations that use the smallest number of product terms. Although output-programmability support is a useful PLD option, many widely used PLDs contain inverting output buffers that are fixed instead of programmable. The examples that follow demonstrate the limitations of PLDs that don't have programmable output polarity.

For instance, **Fig 1** illustrates the architecture for a PLD that uses a single D flip-flop and an inverter in its output stage. **Fig 1a** shows a design that uses an active-high output name, and **Fig 1b** shows one that uses an active-low output name. The pin declarations for **Fig 1a** are

```
PIN 1 = CLOCK
PIN 2 = INPUT
PIN 3 = G
PIN 18 = !OUTPUT.
```

To see why support for output programmability is so important, imagine that the flip-flop's output is fed back to keep it set. The polarity used in the output name makes a significant difference in the number of product (P) terms that are fed back.

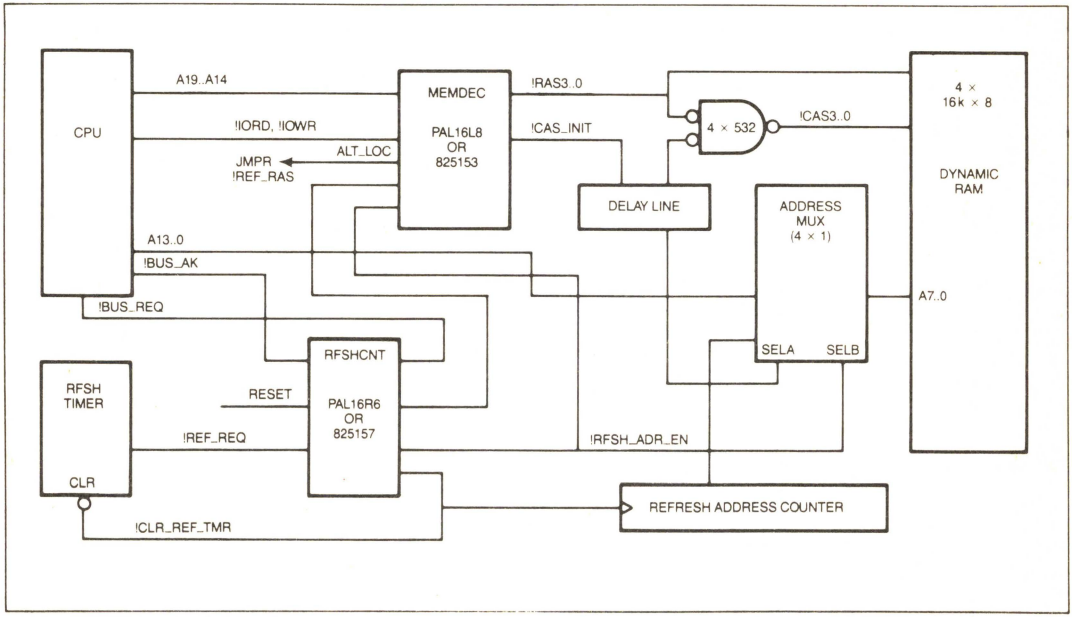



*Fig 3– Memory decoders (MEMDEC) prove a challenging application for PLD conversions. This decoder is a portion of a dynamic RAM controller.*

## TABLE 2 — RFSHCNT LOGIC DESCRIPTION FILE

```
PARTNO      2600A00005 ;
NAME        RFSHCNT ;
DATE        02/19/84 ;
REV         02 ;
DESIGNER    OSANN ;
COMPANY     ASSISTED TECHNOLOGY ;

/*****************************************************************************/
/*   THIS DEVICE RESPONDS TO THE REFRESH REQUEST(REF_REQ) GENERATED          */
/*   BY THE REFRESH INTERVAL TIMER. IT PRODUCES THE SIGNAL WHICH             */
/*   GATES THE REFRESH COUNTER ADDRESS INTO THE RAM ADDRESS BUS              */
/*   AS WELL AS THE REFRESH RAS STROBE AND THE CLEAR PULSE FOR               */
/*   THE REFRESH INTERVAL TIMER.                                             */
/*****************************************************************************/

/** ALLOWABLE TARGET DEVICE TYPES: PAL16R6, 82S157                          **/

/** INPUTS **/
PIN 1           = CLK           ; /* CPU CLOCK */
PIN 2           = REF_REQ       ; /*REFRESH REQUEST FROM INTERVAL TIMER */
PIN 3           = !BUS_AK       ; /* BUS ACKNOWLEDGE FROM CPU */
PIN 4           = RESET         ; /* SYSTEM RESET */
PIN 11          = !OE           ; /* TIED TO GROUND */

/** OUTPUTS **/
PIN 18          = !BUS_REQ      ; /* BUS REQUEST TO CPU */
PIN 17          = !REF_ADR_EN   ; /* ENABLE REFRESH ADDRESS */
PIN 16          = !REF_RAS      ; /* STROBE FOR RAS-ONLY REFRESH */
PIN 15          = !REF_RAS_DLY1 ; /* REF_RAS DELAYED 1 CLOCK */
PIN 14          = !REF_RAS_DLY2 ; /* REF_RAS DELAYED 2 CLOCKS */
PIN 13          = !CLR_REF_TMR  ; /* PULSE TO CLEAR RFRSH INTERVAL TIMER */

/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/
FIELD ST        = [BUS_REQ,     /* ALL OUTPUTS ARE PART OF */
                   REF_ADR_EN,  /* THE STATE BIT FIELD. */
                   REF_RAS,
                   REF_RAS_DLY1,
                   REF_RAS_DLY2,
                   CLR_REF_TMR] ;

/** LOGIC EQUATIONS **/
BUS_REQ.D = !RESET &
            !BUS_REQ & REF_REQ                          /* SET IT */
            # BUS_REQ & ( ST:20 # ST:30                 /* KEEP IT SET */
                        # ST:38 # ST:3C                 /* KEEP IT SET */
                        # ST:3E )       );              /* KEEP IT SET */

REF_ADR_EN.D = !RESET &
               ( !REF_ADR_EN & BUS_AK & BUS_REQ         /* SET IT */
                 # REF_ADR_EN & ( ST:30 # ST:38         /* KEEP IT SET */
                                # ST:3C # ST:3E ) );    /* KEEP IT SET */

REF_RAS.D = !RESET & (ST:30 # ST:38 # ST:3C) ;

REF_RAS_DLY1.D = !RESET & (ST:38 # ST:3C # ST:3E) ;

REF_RAS_DLY2.D = !RESET & (ST:3C # ST:3E # ST:36) ;

CLR_REF_TMR.D = !RESET & ST:36 ;
```

*Conditional control structures improve compiler flexibility. They allow the compiler to delay decisions until certain predefined conditions are met.*

If you choose an active-high output name, the logic equations are

```
OUTPUT.D = G & INPUT /* UPDATE WITH INPUT */
        # !G & OUTPUT /* MAINTAIN CURRENT OUTPUT */
                      /* VIA INTERNAL FEEDBACK PATH */.
```

Because of the inverting output buffer, the equations that you must program into the array are

```
!OUTPUT.D = !(G & INPUT # !G & OUTPUT)
!OUTPUT.D = !G & !OUTPUT # !INPUT & G #
            !INPUT & !OUTPUT.
```

Notice the extra product terms that are created. If, on the other hand, you choose an active-low output name, the pin declarations are

```
PIN 1 = CLOCK
PIN 2 = INPUT
PIN 3 = G
PIN 18 = !OUTPUT.
```

and the final equations are

```
OUTPUT.D = G & INPUT /* UPDATE WITH INPUT */
        # !G & OUTPUT /* MAINTAIN CURRENT OUTPUT */
                      /* VIA INTERNAL FEEDBACK PATH */.
```

As you can see, when PLDs have fixed inverting buffers, the active-low output condition requires the fewest number of P terms.

Now that you're familiar with CUPL's features, you're ready to apply them to more complicated systems. When a logic designer uses a PLD for the first time in a new design, the designer's target area is often the address-decode function. **Fig 2** shows a simple I/O-decoding circuit that creates a buffer-enable signal for I/O reads or writes when the decoded address falls in the hexadecimal range 10 through 12, inclusively. If you were to implement this address-decoding function using assembler-based software, your equations would look like the following ones:

```
BUFFEN=IORD*/A7*/A6*/A5*A4*/A3*/A2*/A1*/A0
      +IORD*/A7*/A6*/A5*A4*/A3*/A2*/A1*A0
      +IORD*/A7*/A6*/A5*A4*/A3*/A2*A1*/A0
      +IOWR*/A7*/A6*/A5*A4*/A3*/A2*/A1*/A0
      +IOWR*/A7*/A6*/A5*A4*/A3*/A2*/A1*A0
      +IOWR*/A7*/A6*/A5*A4*/A3*/A2*A1*/A0.
```

If you were to implement the address-decoding function using CUPL, your equations would look like this:

```
FIELDADR = [A7 . . 0];
IOREQ = IORD # IOWR;
BUFFEN = IOREQ & ADR:[10 . . 12];.
```

To write equations using CUPL, you first define the address bus as a bit field where ADR=[A7 . . . 0]. The compiler then substitutes [A7 . . . 0] whenever it sees ADR. You then combine the strobe signals and give them the arbitrary name IOREQ where IOREQ =IORD#IOWR. Finally, you write an equation for the output BUFFEN in terms of the intermediate variables IOREQ and ADR so that BUFFEN=IOREQ&ADR:[10 . . . 12]. The list-notation and range functions, as well as macro substitution, are all used here. The final code takes less time to write and is much easier to read than code written in an assembler-based language, and it's virtually self documenting.

**Fig 3** shows the CUPL design technique in a more complicated decoder application, a dynamic RAM controller. The PLD MEMDEC in **Fig 3** provides the memory decoder function. It supplies four 16k×8-bit banks of dynamic RAM with RAS (row address strobe) signals and generates a signal that initiates the CAS (column address strobe). The initiating signal first passes through a delay line and then recombines with the RAS signals to produce the CAS.

MEMDEC decodes address bits A19 through A14 of a 20-bit address space and maps the 64k-byte block to either the top or the bottom of the memory map shown in **Fig 4**. The jumper-selectable input called ALT_LOC

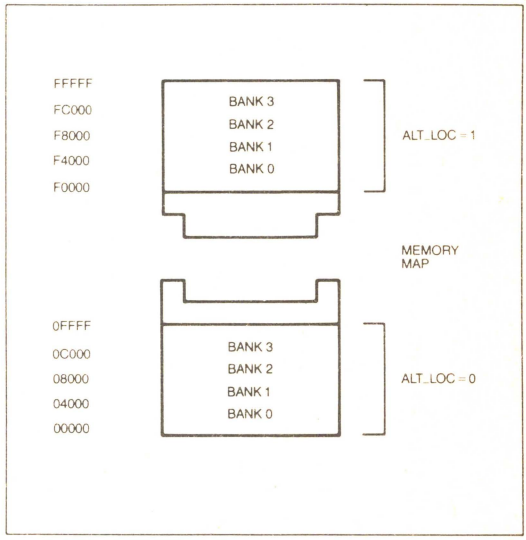



***Fig 4—This memory map shows two possible locations** for address bits A19 through A14 of a 20-bit address space. MEMDEC decodes the bits and maps the 64k-byte block to either the top or the bottom of the memory map.*

*Programmable-output capability allows the compiler to save PLD space. Thus, you'll need fewer PLDs when you're converting your design.*

determines whether the top or the bottom of the memory map is used. **Table 1** shows a completed LDF for the memory decoder.

Not only does CUPL simplify combinatorial designs, but it's useful for implementing sequential designs as well. Because PLDs contain both the logic array and registers in the same package, they're particularly powerful for implementing registered logic. The PLD named RFSHCNT in the RAM controller shown in **Fig 3** handles the sequential aspects of refresh control for the dynamic RAM in a typical μP system.

RFSHCNT responds to a refresh signal from the refresh internal timer (usually 14 μsec) by driving the CPU's bus-request line high. After receiving a bus-acknowledge signal from the CPU, RFSHCNT then generates signals for address MUX control and RAS-only refresh timing.

RFSHCNT also provides a signal that resets the refresh interval timer and clocks the refresh-address counter. **Fig 5** shows the timing diagram for RFSHCNT. Note that the registered output signals are shown as logical true even though the actual outputs are active low. Because the equations are based on signals in the timing diagram, in order for the registered outputs to be shown as logical true, the target device must have either an inverting output buffer or programmable-output-polarity capability.

**Table 2** shows the LDF for RFSHCNT. The LDF uses the hexadecimal values that define the time slots shown in the timing diagram. Note the use of CUPL's bit-field capability in the equations that specify the D flip-flop's state.

CUPL's compiler-based techniques simplify the conversion of complicated SSI circuits to their PLD equivalents. Part 3 of this series will show you how to simplify the logic design process even further by using the state-machine approach. **EDN**

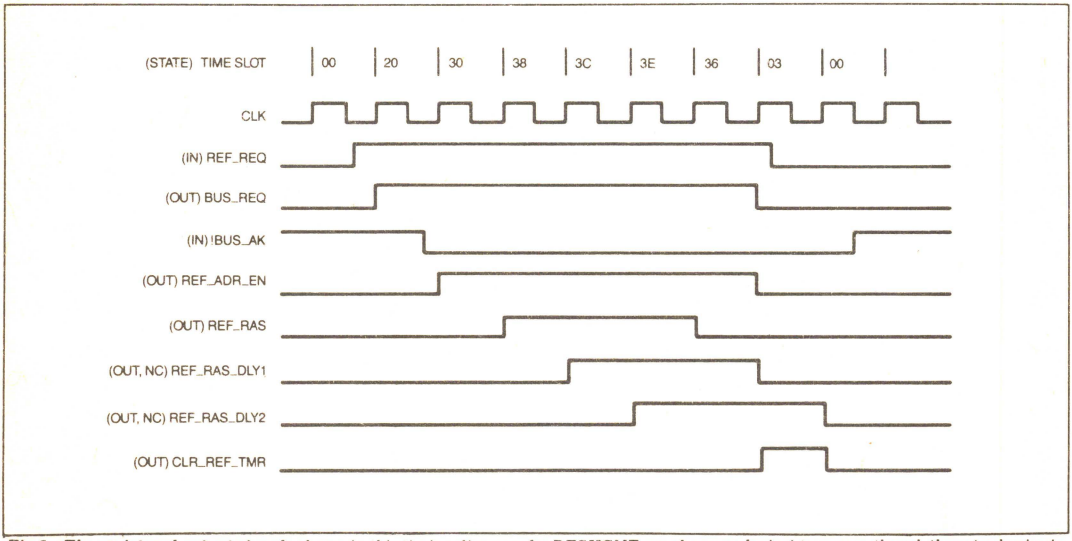



***Fig 5—The registered output signals** shown in this timing diagram for RFSHCNT are shown as logical true even though the actual outputs are active low. Because the equations are based on the timing diagram, the target device must be inverting, or it must have programmable-output capability.*

# CUPL

## State-Machine Approach Speeds Logic Design — Part 3

*To exercise a PLD's full potential for shortening design time and improving documentation, use the state-machine approach. This approach lets you formulate a behavioral description of your system and implement it directly in a PLD, without ever developing an equation-level representation.*

Using the state-machine approach and a compiler-based PLD design language like CUPL, you can bypass the gate- and equation-level stage in logic design and move directly from a system-level description to a PLD implementation. Unlike assembler-based approaches, the state-machine approach lets you document your design in a manner that's understandable to future users of your design.

Actually, few logic designers currently use the state-machine approach in their logic designs. This isn't surprising: The technique seems difficult to learn at first. But CUPL makes the state-machine approach less formidable by handling many of the decisions you would normally have to make. Furthermore, CUPL gives you a general and simple state-machine model like the one shown in **Fig 1.** The software automatically fits the model to your application.

EDN February 7, 1985

### Defining the state model

In general, a state machine is a logic circuit with flip-flops. Because a flip-flop's output can be fed back to its own or some other flip-flop's input, a flip-flop's input value may depend on both its own output and that of other flip-flops. Consequently, the final value for a flip-flop's output depends on its own previous values, as well as those of other flip-flops.

The CUPL state-machine model uses six compo-

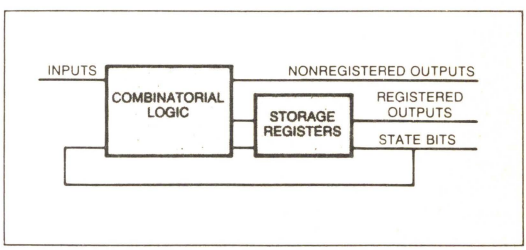



***Fig 1—State-machine theory can be complicated,*** *but CUPL allows you to abstract from the theory's complicated details. Using this simple model and an easy-to-learn syntax, you can quickly construct state-machine models of your system.*

*When you use the state-machine approach, you don't have to write a logic-equation-level description of your system before implementing it in a PLD.*

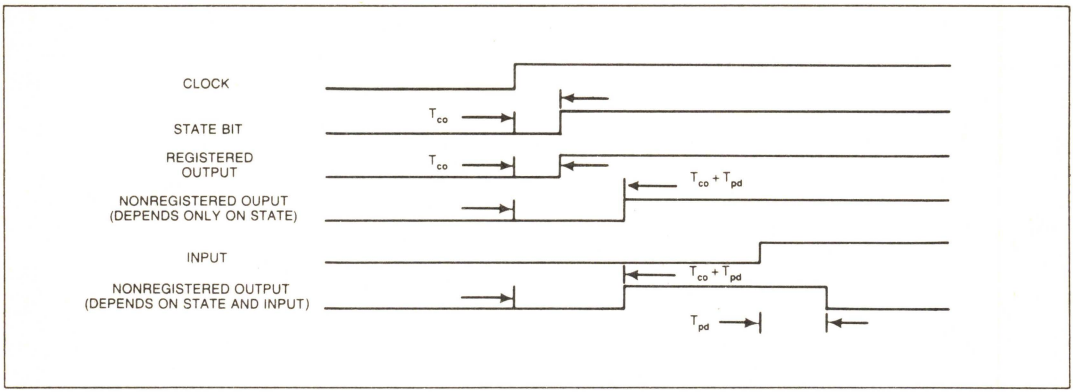



***Fig 2—This timing diagram*** *characterizes CUPL's simple state-machine model. The setting or resetting of the registered output depends on the status of the state bit. Conversely, nonregistered outputs can depend either on only the current state bit's status or on both the state bit's status and the input's status.*

nents: inputs, combinatorial logic, storage registers, state bits, registered outputs, and nonregistered outputs. **Fig 2** shows the timing relationships between these components.

Inputs are signals entering the device that originate in some other device. Combinatorial logic is any combination of logical gates (usually AND-OR) that produces an output signal that's valid $T_{pd}$ (propagation delay time) nsec after any of the signals that drive these gates changes. $T_{pd}$ is the time delay between the initiation of an input or feedback event and the occurrence of a nonregistered output.

State bits are storage-register outputs that are fed back to drive the combinatorial logic. They contain the present-state information. Storage registers are any flip-flop elements that receive their inputs from the state machine's combinatorial logic. Some registers are used for state bits, while others are used for registered outputs. The registered output is valid $T_{co}$ nsec after the clock pulse occurs. $T_{co}$ is the time delay between the initiation of a clock signal and the occurrence of a valid flip-flop output.

For the system to operate properly, you must meet your PLD's requirements for setup and hold times. For most PLDs, the setup time ($T_{su}$) usually includes both the propagation delay of the combinatorial logic and the actual setup time of the flip-flops. $T_{su}$ is the time it takes for the result of either a feedback or an input event to appear at the input to a flip-flop. A subsequent clock input cannot be applied until this result becomes valid at the flip-flop's input. These flip-flops may be either D, RS, or JK types (but RS and JK types are used more often in state-machine implementations because they require fewer product (P) terms than D types do).

Nonregistered outputs are outputs that come directly from the combinatorial logic gates. They may be functions of the state bits and the input signals (and have asynchronous timing), or they may be purely dependent on the current state-bit values, in which case they become valid $T_{co}+T_{pd}$ nsec after an active clock edge occurs.

Registered outputs are outputs that come from the storage registers but are not included in the actual state-bit field (ie, a bit field composed of all the state bits). State-machine theory requires that the setting or resetting of these registered outputs depend on the *transition* from a present state to a next state. This allows a registered output to be either set or reset in a given state, depending on how the machine came to be in that state. Thus, a registered output can assume a "don't care" operation mode. In the "don't care" mode, the registered output will remain at its last value as long as the current state transition does not specify that registered output.

### The state-machine syntax

To help you implement this state-machine model quickly, CUPL supplies a general and simple state-machine syntax. This syntax gives you a single, simple format that allows you to describe any function in the state machine. The general format for the state-machine syntax is

```
SEQUENCE state_bit_field {
  PRESENT present_state
    IF input_cond   NEXT next_state   OUT outputs ;
    IF input_cond   NEXT next_state   OUT outputs ;
    IF . . .
  PRESENT present_state
    IF input_cond   NEXT next_state   OUT outputs ;
    IF input_cond   NEXT next_state   OUT outputs ;
    IF . . .
  PRESENT . . .
}
```

Each present-state block within this format describes both asynchronous (present state) and synchronous (transition) activity. Using this format, you can describe any component of the state machine. For example, the formats for registered outputs would be

| IF input_cond | NEXT next_state | OUT outputs |
|---|---|---|
| CONDITIONAL TRANSITION | | OUTPUT ASSOCIATED WITH TRANSITION |

or

| | NEXT next_state | OUT outputs |
|---|---|---|
| UNCONDITIONAL TRANSITION | | OUTPUT ASSOCIATED WITH TRANSITION, |

depending on whether the transition is conditional or not. To use these equations for describing your system, you need to learn how to use the CUPL keywords. For example, when you use a Next statement, you're telling the compiler that all of the outputs in that block are registered outputs whose values depend on transition information (ie, information about the transition from the present state to the next state). Using the If statement signifies a conditional event. When you use the If keyword in a nonregistered description, you signify that the input and output events will have an asynchronous dependence. The absence of a Next keyword signifies a nonregistered event.

For nonregistered outputs, you would use the format

| IF input_cond | | OUT outputs |
|---|---|---|
| INPUT CONDITION AFFECTS OUTPUT AFTER $T_{pd}$ | NO STATE TRANSITION | OUTPUTS ASSOCIATED WITH INPUT CONDITION AND PRESENT STATE. |

or

| | | OUT outputs |
|---|---|---|
| NO INPUT CONDITION | NO STATE TRANSITION | OUTPUT ASSOCIATED SOLELY WITH PRESENT STATE. VALID $T_{co}$ + $T_{pd}$ AFTER CLOCK. |

Much of the reason for choosing either the registered or nonregistered format for an output depends on the system timing. For fully synchronous systems that require tight timing, the registered output provides fast response—it responds within $T_{co}$ nsec after the occurrence of a clock pulse. This quick response gives the circuit time to use that registered output as an input somewhere else in the circuit before the next clock pulse occurs.

Conversely, you would use the nonregistered output in asynchronous applications. You would also use the

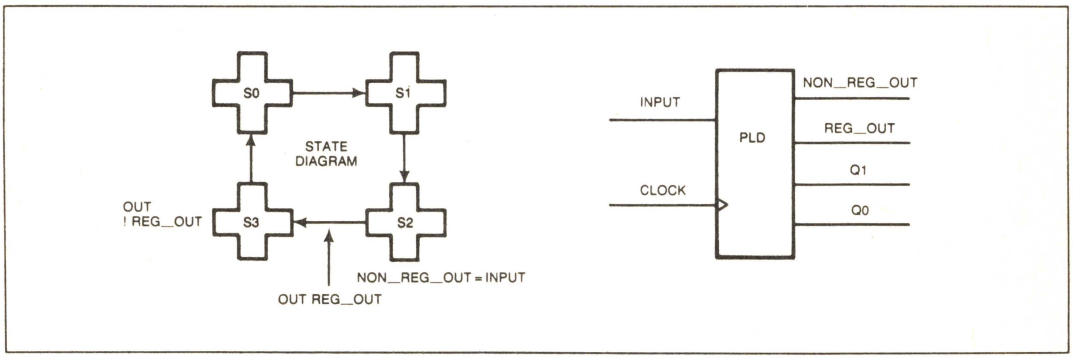



***Fig 3—This model for a free-running 2-bit counter*** *demonstrates CUPL's state-machine syntax. The counter has one input, one nonregistered output, and one registered output.*

nonregistered output in simpler applications, such as present-state decoders.

To better understand the state-machine model and its syntax, consider a simple example: a free-running 2-bit counter with one input, one registered output, and one nonregistered output. **Fig 3** shows the state-transition diagram. The circles represent states (specific combinations of the state bits), and the arrows represent the transitions between states. Because the transitions in this example are unconditional, the counter is free-running. Accordingly, the logic description uses no If keywords in statements that signify a Next state. The

## The function-table approach

To design logic systems with PLDs, you could use the function-table approach, which complements the state-machine approach. The function-table approach is useful in applications such as code converters, where input/output relationships are best represented in tabular form.

CUPL's parallel-operation capability makes it easy for you to develop these tabular representations. Using that feature, you can declare bit fields and use them on either the right or left side of the equation.

The parallel operation feature allows you to operate uniformly

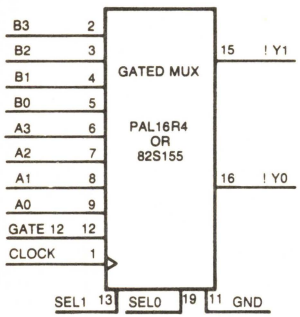



***Fig A—In code-conversion applications,*** *like this dual 4-to-1 multiplexer, you can best describe the system using a tabular format.*

### TABLE A—GATED MUX LOGIC DESCRIPTION FILE

```
PARTNO      PL10007;
NAME        GATED MUX;
DATE        09/17/84;
REV         01;
DESIGNER    ARONSON;
COMPANY     ASSISTED TECHNOLOGY;
ASSEMBLY    PC_IO;
LOCATION    U23;

/*************************************************************************/
/* THIS DEVICE FUNCTIONS AS A DUAL 4-TO-1 MUX WITH INVERTING             */
/* REGISTERED OUTPUTS. THE MUX OUTPUTS ARE ONLY CLOCKED INTO THE         */
/* REGISTERS WHEN THE GATE INPUT IS ACTIVE.                              */
/*************************************************************************/
/* ALLOWABLE TARGET DEVICE TYPES : PAL16R4 , 82S155                      */
/*************************************************************************/


/** INPUTS **/

PIN 1       = CLOCK     ; /* SYSTEM CLOCK */
PIN [2..5]  = [B3..0]   ; /* INPUT GROUP B */
PIN [6..9]  = [A3..0]   ; /* INPUT GROUP A */
PIN 13      = SEL1      ; /* SELECT 1 */
PIN 19      = SEL0      ; /* SELECT 0 */
PIN 12      = GATE      ; /* GATES MUX OUTPUT INTO REGISTER */
PIN 11      = !OE       ; /* OUTPUT ENABLE */


/** OUTPUTS **/

PIN 15      = !Y1       ; /* REGISTER OUTPUT FROM GROUP B */
PIN 16      = !Y0       ; /* REGISTER OUTPUT FROM GROUP A */


/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/

FIELD OUT = [Y1..0] ;   /* OUTPUT BITS */
FIELD SEL = [SEL1..0] ; /* SELECT CONTROL BITS */

/** LOGIC EQUATIONS **/

OUT.D = !GATE & OUT                 /*****************************/
      # GATE & ( [B3,A3] & SEL:3    /* NOTE:                     */
               # [B2,A2] & SEL:2    /*   ONE EQUATION DESCRIBES  */
               # [B1,A1] & SEL:1    /*   BOTH OUTPUT VARIABLES.  */
               # [B0,A0] & SEL:0 ) ; /*****************************/
```

*To make the state-machine approach easier to learn, CUPL uses a model that incorporates both the Mealy and Moore models. You use the same model for all cases.*

nonregistered output is active on a count of two (S2) when the input is active. The registered output is set on the transition from S2 to S3 and reset on the transition from S3 to S0. **Table 1** gives the logic description for the counter.

An application that incorporates hysteresis shows the importance of using transition information in addition to present-state information. Consider, for instance, a circuit that performs threshold detection on an analog signal, but requires a hysteresis band both wider and more accurate than the hysteresis band an analog comparator could achieve. In such an application, you

## TABLE B—HEXDISP LOGIC DESCRIPTION FILE

```
PARTNO      CT0002;
NAME        HEXDISP;
DATE        6/5/84;
REV         01;
DESIGNER    T KAHL;
COMPANY     ASSISTED TECHNOLOGY INC;
ASSEMBLY    DISPLAY_BOARD;
LOCATION    U17;

/*************************************************************************/
/* THIS IS A HEXADECIMAL-TO-SEVEN-SEGMENT                                */
/* DECODER CAPABLE OF DRIVING COMMON-ANODE                               */
/* LEDS. IT INCORPORATES BOTH A RIPPLE-                                  */
/* BLANKING INPUT (TO INHIBIT DISPLAYING                                 */
/* LEADING ZEROES) AND A RIPPLE-BLANKING                                 */
/* OUTPUT TO ALLOW FOR EASY CASCADING OF                                 */
/* DIGITS.                                                               */
/*                                                                       */
/*                                                                       */
/*                                                                       */
/*************************************************************************/
/* ALLOWABLE TARGET DEVICE TYPES: 32 x 8 PROM (82S123 OR EQUIV)          */
/*************************************************************************/

/** INPUTS **/

PIN [10..13] = [D0..3] ;        /* DATA INPUT LINES TO DISPLAY          */
PIN 14 = !RBI;                  /* RIPPLE BLANKING INPUT                */

/** OUTPUTS **/

PIN [7..1] = ![A,B,C,D,E,F,G];  /* SEGMENT OUTPUT LINES                 */
PIN 9      = !RBO;              /* RIPPLE BLANKING OUTPUT               */

/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/

FIELD DATA = [D3..0];           /* HEXADECIMAL INPUT FIELD              */
FIELD SEGMENT=[A,B,C,D,E,F,G] ; /* DISPLAY SEGMENT FIELD                */

$DEFINE ON  'b'1                /* SEGMENT LIT WHEN LOGICALLY "ON"      */
$DEFINE OFF 'b'0                /* SEGMENT DARK WHEN LOGICALLY
                                   "OFF"                                */

/** LOGIC EQUATIONS **/

                /*  A    B    C    D    E    F    G   */
SEGMENT =
/* 0 */         [ ON,  ON,  ON,  ON,  ON,  ON,  OFF] &  DATA:0 & !RBI
/* 1 */      #  [OFF,  ON,  ON, OFF, OFF, OFF, OFF] &  DATA:1
/* 2 */      #  [ ON,  ON, OFF,  ON,  ON, OFF,  ON] &  DATA:2
/* 3 */      #  [ ON,  ON,  ON,  ON, OFF, OFF,  ON] &  DATA:3
/* 4 */      #  [OFF,  ON,  ON, OFF, OFF,  ON,  ON] &  DATA:4
/* 5 */      #  [ ON, OFF,  ON,  ON, OFF,  ON, OFF] &  DATA:5
/* 6 */      #  [ ON, OFF,  ON,  ON,  ON,  ON, OFF] &  DATA:6
/* 7 */      #  [ ON,  ON,  ON, OFF, OFF, OFF, OFF] &  DATA:7
/* 8 */      #  [ ON,  ON,  ON,  ON,  ON,  ON,  ON] &  DATA:8
/* 9 */      #  [ ON,  ON,  ON, OFF, OFF,  ON,  ON] &  DATA:9
/* A */      #  [ ON,  ON,  ON, OFF,  ON,  ON,  ON] &  DATA:A
/* B */      #  [OFF, OFF,  ON,  ON,  ON,  ON,  ON] &  DATA:B
/* C */      #  [ ON, OFF, OFF,  ON,  ON,  ON, OFF] &  DATA:C
/* D */      #  [OFF,  ON,  ON,  ON,  ON, OFF,  ON] &  DATA:D
/* E */      #  [ ON, OFF, OFF,  ON,  ON,  ON,  ON] &  DATA:E
/* F */      #  [ ON, OFF, OFF, OFF,  ON,  ON,  ON] &  DATA:F;

RBO = RBI & DATA.0;
```

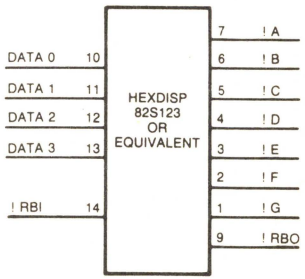



***Fig B—Though a PROM was used to implement this simple hexadecimal-to-7 segment decoder,*** *you could use the same function table to implement this state machine in a PLD.*

on an entire parallel data path. It's easy to write a description in this manner, as you can see from **Fig A**'s 4-to-1 multiplexer, which has an inverting registered output. **Table A** contains the multiplexer's LDF.

The hexadecimal-to-7 segment decoder in **Fig B** also lends itself well to tabular representation. Again, bit-field notation is convenient for describing the logical function. Incidentally, because it's a simple circuit, this implementation uses a PROM as the target device (because 16 pins are sufficient and bipolar PROMs are inexpensive), but you could also implement these function tables in PLDs. **Table B** contains the decoder's LDF.

## TABLE 1—LOGIC DESCRIPTION FOR 2-BIT COUNTER

```
FIELD COUNT = [Q1, Q0];        /* LET'S CALL THE STATE BIT FIELD "COUNT" */

                /*************************************************************/
$DEFINE S0 0    /*DEFINE SYMBOLIC NAMES FOR THE ACTUAL STATE BIT CONSTANT    */
$DEFINE S1 1    /* VALUES USING PREPROCESSOR COMMANDS. CONSTANTS DEFAULT     */
$DEFINE S2 2    /*      TO HEX AND REPRESENT VALUES OF "COUNT" WITHIN THE     */
$DEFINE S3 3    /*                  "SEQUENCE" BLOCK BELOW.                   */
                /*************************************************************/

SEQUENCE COUNT {    /*      NOTE USE OF BRACES FOR ENCLOSING STATE           */
                    /*           SEQUENCE DESCRIPTION BLOCK.                 */

PRESENT S0
                NEXT S1;
PRESENT S1
                NEXT S2,
PRESENT S2      IF INPUT OUT NON_REG_OUT;  /*   ASYNCHRONOUS WITHIN S2       */
                NEXT S3 OUT REG_OUT;       /*     SETS ON TRANSITION         */
PRESENT S3
                NEXT S0 OUT !REG_OUT ; }   /*     RESETS ON TRANSITION       */
```

need transition information in order to achieve hysteresis. One way to solve this problem would be to construct a tracking A/D converter in which the threshold detector output (digital Schmitt-trigger output) is a registered output of the state machine (**Fig 4**).

The three counter bits that feed the D/A converter compose the state bits. To create the hysteresis, you set the trigger output only on the transition from S5 to S6 and reset the trigger only on the transition from S2 to S1. At all other times, you place the trigger output in a "don't care" state. The trigger output may have different values in states S2 through S5 depending on how the machine arrived at those states.

**Fig 5** shows a state diagram for the system. All states in which you can set the trigger output are shown on top and all states in which you can reset the trigger

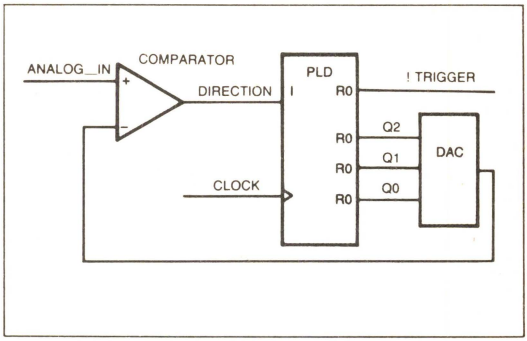



***Fig 4—To realize the hysteresis function*** *in this state-machine model for an analog comparator with digital hysteresis, you must set or reset the registered output by using transition information rather than present-state information.*

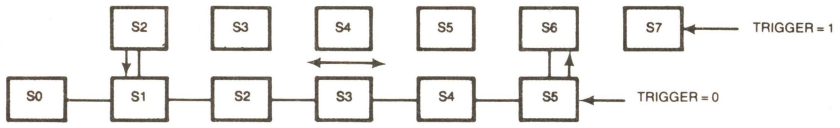



***Fig 5—The state diagram for the analog comparator with hysteresis*** *shows that a state's value can be history-dependent. The trigger output can have different values in states S2 through S5 depending on how the machine arrives at those states.*

*The CUPL state-machine syntax allows you to specify any state-machine component with a single format, thus simplifying the state-machine description.*

## TABLE 2—SCHMITT LOGIC DESCRIPTION FILE

```
PARTNO      CT0001;
NAME        SCHMITT;
DATE        6/30/84;
REVISION    01;
DESIGNER    T KAHL;
COMPANY     ASSISTED TECHNOLOGY INC;
ASSEMBLY    ANALOG  INTERFACE;
LOCATION    U27;

/*************************************************************************/
/* THIS DEVICE RECEIVES A 'COUNT DIRECTION' COMMAND FROM AN ANALOG       */
/* COMPARATOR AND RESPONDS BY INCREMENTING OR DECREMENTING AN            */
/* INTEGRAL UP/DOWN COUNTER. A REGISTERED OUTPUT IS CREATED AND ACTS     */
/* AS A DIGITAL SCHMITT TRIGGER WITH HYSTERESIS.                         */
/*************************************************************************/


/** INPUTS **/

PIN 1           = CLOCK;        /* CLOCK PIN FOR THE COUNTER    */
PIN 2           = DIRECTION;    /* DIRECTION OF COUNT MODE PIN  */

/** OUTPUTS **/

PIN [14..16]    = ![Q0..2];     /* COUNTER STATE BITS           */
PIN 17          = !TRIGGER;     /* SCHMITT TRIGGER OUTPUT BIT   */

/** DECLARATIONS AND INTERMEDIATE VARIABLE DEFINITIONS **/
UP = DIRECTION;                 /* COUNTER MODES                */
DOWN = !DIRECTION;

FIELD COUNT = [Q2..0];          /* FIELD FOR COUNTER STATES     */

$DEFINE S0 0                    /* COUNTER STATES DEFINED AS    */
$DEFINE S1 1                    /* STATES 0 THRU 7              */
$DEFINE S2 2
$DEFINE S3 3
$DEFINE S4 4
$DEFINE S5 5
$DEFINE S6 6
$DEFINE S7 7


SEQUENCE COUNT {

PRESENT S0
                IF UP           NEXT S1;
                IF DOWN         NEXT S0;
PRESENT S1
                IF UP           NEXT S2;
                IF DOWN         NEXT S0;
PRESENT S2
                IF UP           NEXT S3;
                IF DOWN         NEXT S1         OUT !TRIGGER;
PRESENT S3
                IF UP           NEXT S4;
                IF DOWN         NEXT S2;


PRESENT S4
                IF UP           NEXT S5;
                IF DOWN         NEXT S3;
PRESENT S5
                IF UP           NEXT S6         OUT TRIGGER;
                IF DOWN         NEXT S4
PRESENT S6
                IF UP           NEXT S7;
                IF DOWN         NEXT S5;
PRESENT S7
                IF UP           NEXT S7;
                IF DOWN         NEXT S6;
```

are shown on the bottom. Note that states S2, S3, S4, and S5 appear twice because they can have two different values. Each state's value depends on the system's previous state.

Note also that the state bits in this application supply information to the outside world; in this case, the information consists of inputs to a D/A converter. When you give the PLD access to the outside world, you deviate from the standard Mealy and Moore state-machine models, but you can squeeze more logic into your PLD.

**Table 2** gives the state machine's logic description file (LDF). In the LDF, you declare the state bits as a bit field and give them the symbolic name "Count." Next, you use the input Direction to define names for the Up and Down counter modes. You then complete the numerical state assignment for states S0 through S7 by using the $Define command from CUPL's preprocessor.

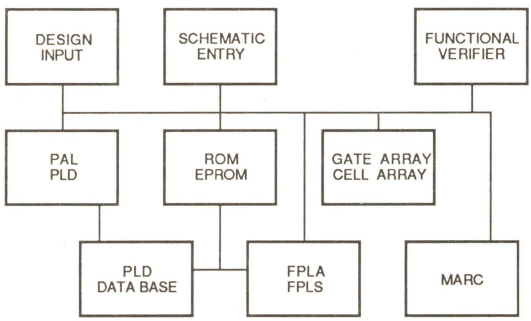
In defining the state machine, you use If and Next keywords for every present-state block. When you use Next, you indicate that the state machine's activity is synchronous; when you use If, you indicate that the transitions are conditional. The transitions' direction depends on the direction the counter counts in, which is in turn determined by the value of the Direction input.

Though applications like counters and comparators with hysteresis may not seem very complicated, they serve to show that designing with PLDs is a straightforward task, whether you're using the devices to replace existing designs or using them in a state-machine design. **EDN**

# LOG/iC

## LOG/iC: The CAE System for Digital Electronics

- **A modular design tool for digital electronics for devices from the entire ASIC spectrum.**
- **Supports all types of PLDs, regardless of manufacturer.**
- **Extensive library, automatic generation of test vectors.**
- **Development of multi-level gate circuits for gate arrays, cell arrays, etc., including a timing analysis and the processing of net lists.**
- **Complete optimization even for MARCs (microprogrammed designs), state machines of very high complexity.**
- **Same standardized input syntax for all devices, making it easy to switch between different implementations of a circuit. The input data set is not restricted to describe one device and may contain larger circuits.**
- **Specific optimization strategies for every device family. Exact minimization results reached in unequaled speed.**
- **Standard data format for the programming and test data for PLDs. Communication software facilitates a trouble-free connection with the programmer. Interfaces to the CAE systems of many manufacturers.**
- **Can be run on various mainframes (e.g. VAX), workstations (e.g. Apollo), and PCs (e.g. IBM-PC).**




418 01

### Functional Description

LOG/iC is a modular design tool for the development of digital circuits. The system is based on a design specification formulated in its own versatile syntax. In addition, it allows schematic entry of third-party systems via net lists. Circuit definitions gained in this way are checked and optimized specifically for each device. LOG/iC transforms this optimized circuit into various devices selected from the entire ASIC spectrum, generates test aids and produces specific documentation for each realization of the circuit.

The entire CAE system consists of module packages tailored to various applications and additional options. All module packages use the same input facilities, but they have specific tools to optimize the circuit according to the supported device family.

Module package 1 supports the design of PAL devices. It is also contained in module package 2 which supports all PLDs such as PROMs, PAL devices, and PLS devices. Module package 3 is the tool used to develop multi-level gate circuits implemented in semi- or full-custom circuits. Module package 4 develops so-called MARCs (Minimized Address space ROM-based Controllers).

Optional interfaces allow the connection of "Schematic Entry Systems". The "Functional Verifier" option allows the interactive simulation of a circuit on the functional level, so that there is no need to run a complete compilation in order to verify its logic behavior.

Option "PLD Data Base" rounds out the range of tools offered by LOG/iC. This databank contains the essential data of all PLDs supported by LOG/iC, such as structures, electrical and AC characteristics, etc. Combined with module packages 1 and 2, it supports the selection of the best-suited PLDs.

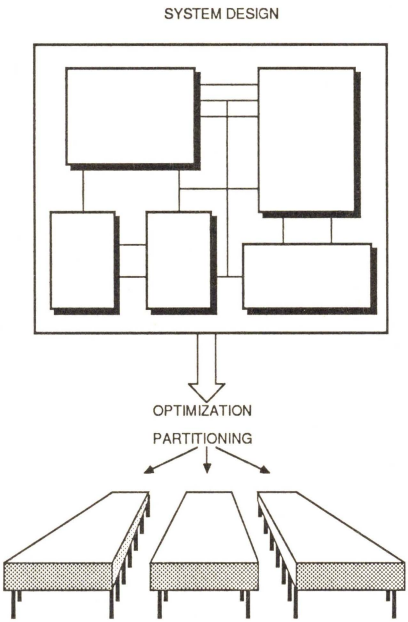
This data sheet describes the basic features of the system. Module packages and options are described in detail on separate data sheets.

## Process of Development

LOG/iC is a universal design system for digital logic. It puts the emphasis on the logical description of the circuit rather than a specific device. LOG/iC allows you to run sophisticated designs without tying you down to a specific IC.

Circuit descriptions, as well as syntax and consistency checks, can be run without transforming the design into a specific device. This applies also to the circuit simulation performed by the "Functional Verifier", which verifies the logical functioning of the design. There is a specific optimization for each particular device family. The designer's choice of a device is based upon the results of this optimization. Additional support is offered by the "PLD Data Base" that helps select the best-suited type in case of a PLD implementation.

The described procedure allows the user to enter and optimize large designs even if they can't be implemented in one single IC. After optimization, LOG/iC helps to select the best-suited circuit technology and IC type, thus enabling the designer to make best use of the benefits of modern ICs.




418 02

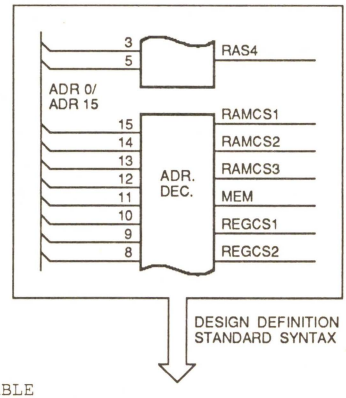
## Circuit Definition

LOG/iC offers specific description aids for various design problems. Its standard syntax preferably describes combinatorial circuits, but it is also possible to describe sequential circuits by means of Boolean equations. The FSM (Finite State Machine) syntax has been created to make the definition of state machines more convenient. Special interface options enable circuits from third-party systems to be fed into LOG/iC in the form of circuit schematics. Together with LOG/iC, any text editor can be used to edit design files.

## Standard Syntax

The basic elements of the standard syntax are Boolean equations and function tables. The format of the equations corresponds to the common standard. All common operators can be used. The equations can be nested in any order. They will automatically be converted into a Sum-of-Products (SOP) format. String substitutions considerably reduce the time spent on editing, and allow the design to be more clearly laid out.

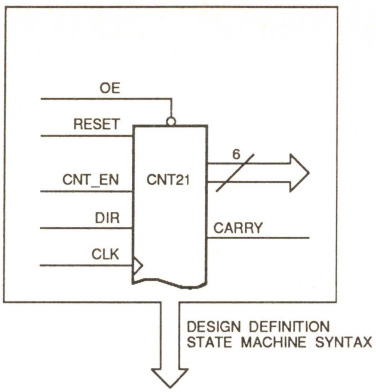
Function tables, basically the most concise way of formatting, can be made even more concise through numbers in arbitrary numeric systems, which can then be merged (hex, decimal, octal, binary). It is possible to partition logical data into a subset of different function tables. In addition, equations and tables can be merged in order to define a circuit. The option "Rest-Definition" and the use of number fields make the function tables particularly concise. As a result, address decoders especially can be conveniently described.




```
*FUNCTION TABLE
$  (ADR[15..0]) :MEM, (RAMCS [1..3]), (REGCS [1..4]);
 ;-------------------------------------------------------
 0000H . . 3FFFH: 1 ,  100 , 0000 ; MEMORY NR.1
 4000H . . 7FFFH: 1 ,  010 , 0000 ; MEMORY NR.2
 8000H . . BFFFH: 1 ,  001 , 0000 ; MEMORY NR.3
     C037H      : 0 ,   -  , 1000 ; S-REGISTER
     CC03H      : 0 ,   -  , 0100 ; F-REGISTER
     D0F7H      : 0 ,   -  , 0010 ; G-REGISTER
     EFF3H      : 0 ,   -  , 0001 ; L-REGISTER
 REST           : 0 ,   -  , 0000 ; NOT SELECTED

 *END
```

418 03




```
* FLOW TABLE
;-------------------------------------------------------
S [1..21], 'CLEAR', 'CAR_0', F1; SYNCHRONOUS RESET

;COUNT UP
S [1..20], 'UP'   , 'CAR_0', F[2..21];
      S21, 'UP'   , 'CAR_1', F1;

;COUNT DOWN
S [21..2], 'DOWN' , 'CAR_0', F[20..1];
      S 1, 'DOWN' , 'CAR_1', F 21;
;-------------------------------------------------------
*STATE ASSIGNMENT
 BINARY
*END
```

418 04

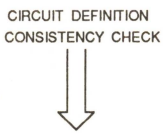
## FSM-Syntax

The FSM-syntax enables you to give a functional description of a synchronous state machine. LOG/iC supports the design of any state machine such as MEALY or MOORE or combinations of the two. Any diagram description can be used as a concept, e.g. flow diagrams or bubble diagrams. These diagrams can easily be transformed into LOG/iC syntax. The basic entry syntax is close to hardware. A number of syntax aids, however, enable circuits to be defined on a high level. Thus you can define input and output vectors and make the design files more understandable by means of the macro-definition. The "range notation" also applies to state definitions so that counters and similar circuits can be conveniently defined. It is possible to select with a single statement the relevant variables out of a whole set of input variables.

## Consistency Check

In addition to the usual syntax check, LOG/iC verifies the logical consistency. At a very early stage of the design flow, inconsistencies are pointed out to the designer. The Consistency Checker reports its results in the form of information, warnings, and errors indicating the line where they occur.

Incompletely specified branches of a state or incomplete function tables are indicated as a warning. Ambiguous outputs of a table, on the other hand, as well as contradictory branches and outputs of a state machine are indicated as an error. The LOG/iC consistency check detects even complex error conditions, such as states that can't possibly be reached by running the circuit.



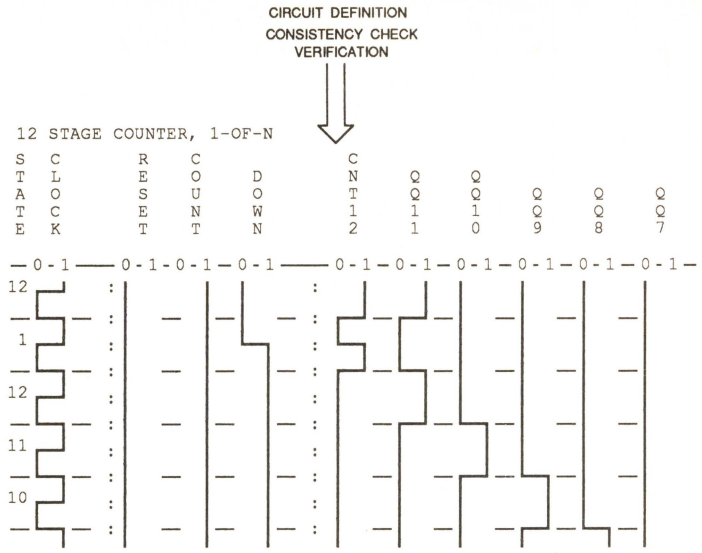

```
**  WARNING  ** STATE    1 INCOMPLETELY DEFINED
**  WARNING  ** STATE    4 INCOMPLETELY DEFINED
****  INFO ****   NO EXIT FROM STATE  4
****  INFO **** STATE    3 CANNOT BE REACHED FROM INITIAL STATE
****  INFO **** STATE    4 CANNOT BE REACHED FROM INITIAL STATE

INCONSISTENT NEXT-STATE ENTRIES IN LINE      1 AND LINE   2

INCONSISTENT CONTROL-VECTORS IN LINE      7 AND LINE   8

*** LOG/iC ERROR TERMINATION: CONSISTENCY-CHECK ***
```

418 05




418 06

## Verification

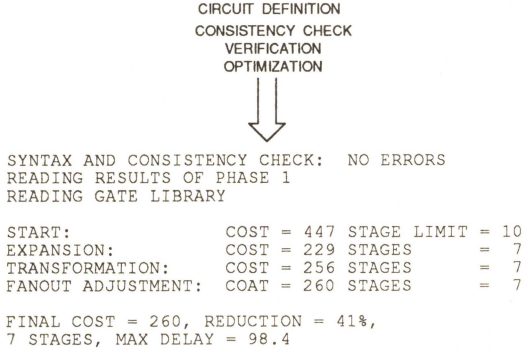
The option "Functional Verifier" is a simulator that applies stimuli to the circuit definition in order to verify its correct logical behavior. This simulation is not influenced by any devices and it provides a quick verification of the design definition. Its operation, similar to a Logic State Analyzer, is screen-based and therefore easy to run. The results of such a simulation are indicated in the form of waveforms. Inputs can be changed online in the interactive mode and the reaction of the simulated circuit will be indicated immediately.

## Optimization

The data input and consistency checks are completely independent of the device. Optimization, however, is tailored to each specific family of devices. Even during optimization, there is still no need to specify the type. PAL device designs are optimized by means of an exact procedure, called the "FACT Algorithm". PLA circuits, on the other hand, are optimized by means of a "bundle"-minimization. Multi-level gate logic for gate and cell arrays is optimized through a special procedure. For ROM-based controllers, the optimizer computes various possible solutions.

By offering a choice of optimizers, LOG/iC enables the user to optimize a defined circuit for very different realizations in a short period of time. The optimization reports produced in this way are tailored to specific device families. They are decision aids for the designer when he wants to realize a particular circuit.

All the optimizers are described in more detail on their respective data sheets.




418 07

```
                                CIRCUIT DEFINITION
                              CONSISTENCY CHECK
                                 VERIFICATION
                                OPTIMIZATION
                                REALIZATION
                                     ⇩
PAL-TYPE: PAL16RP6

                  R                                                  S
                  E          S                       T               M
                  S     T M                          U     M M M     F
                  E N   F F   T T  T T  T T  T T  E A  F F F N  S 1 2 2
                  T C   G 2   A K  Z L  O M  R Z  1 1  2 C  R 6 8 0 T
                  ------------------------------------------------------
                  0 0   0 0   0 1  1 1  1 1  2 2  2 2  2 3
ROW    ADDRESS    0 2   4 6   8 0  2 4  6 8  0 2  4 6  8 0
                  ------------------------------------------------------
                                                                    SMF2
  8     00256     ----  ----  ----  --X-  ---X  ----  ----  X---
  9     00288     ----  ----  ----  ----  X-X-  ----  ----  ----
 10     00320     ----  ----  --X-  ----  ----  X--X  ---X  ----
                                                                    TK
 16     00512     X---  ----  -X-X  ----  ----  -X--  ----  ----
 17     00544     X---  ----  ---X  --X-  ----  ----  ----  ----
 18     00576     X---  -X--  ---X  ----  ----  ----  ----  ----
 19     00608     X---  ----  ----  ---X  --X-  X---  ----  ----
```

418 08

## Realization

In the case of a PLD design, LOG/iC fits the optimized data into the device structure of a particular device and produces the programming and test data for the IC. The structural data are taken from the PLD library and the results are transferred to the programmer in the form of a JEDEC File.

Module package "Gates" produces the net lists of the design in various data formats so that it can be transferred to different CAE systems.

If a design is to be realized as a ROM-based controller, the designer chooses one of the seven solutions offered by the optimizer. The necessary programming and structural data are then produced. LOG/iC provides the programming data for PROMs in the Intel-Hex format.

In all module packages, partial designs can be combined in order to be realized in one circuit. But at the same time, LOG/iC also enables the user to partition a design into various devices.

## Test Aid

LOG/iC is an effective tool for the design of circuits, but it also offers many test aids.

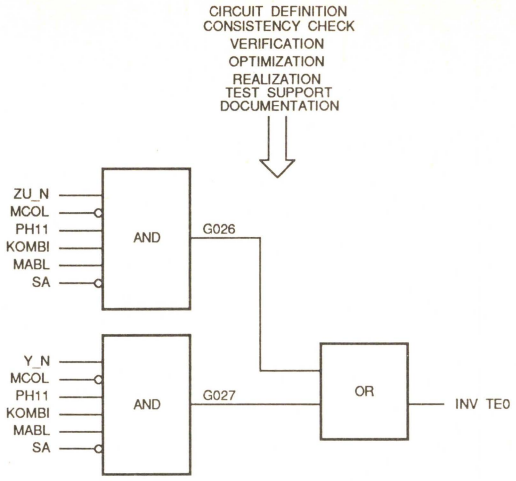
For PAL devices, module package 2 automatically generates test vectors for a product-term-oriented test on the programmer. LOG/iC guarantees 100% product term coverage. Module package 1 supports the input of user-defined test vectors and includes them into the JEDEC File.

Gate array circuits undergo a timing analysis after optimization. The results are extensively documented and contain a critical-path analysis. In addition to the microprogam listing, LOG/iC generates additional test lists for MARCs.

```
                                CIRCUIT DEFINITION
                              CONSISTENCY CHECK
                                 VERIFICATION
                                OPTIMIZATION
                                REALIZATION
                              TEST SUPPORT
                                     ⇩
TEST VECTORS:

1: ;
1: ;          C          R
1: ;          A          E
1: ; C QQ QQR    O G G S          V
1: ; LXQQ QQR    EXN NXEM LRRL MMXC
1: ; K112 34YB AN2D D3TO OOUU UM4C
1: ;-----------------------------
1:   PN11 11NN NNNN PNNN NNNN NNNN ;
2:   01HH HHN0 101N 011H HNNH NL1N ;
3:   C1HH HHH0 101N C11N NNNN NN1N ;  1
4:   PN00 01NN NNNN PNNN NNNN NNNN ;
5:   00LL LHN1 000N 001L LHNN HN0N ;
6:   C0LL LLH1 000N C01N NNNN NN0N ;  2
```

418 09




418 10

## Documentation

Each run of the compiler will be documented in several listings. Those listings are laid out according to the selected device family. The format of the documents can be determined by the user.

Optimization Reports give information on the number of product terms used and the achieved reduction. LOG/iC generates Pinouts, Gate Plots and Circuit Block Diagrams. You can, of course, get the result in the form of Boolean Equations.

Fuse Plots (PAL device specific documents) come in the common notation; in the case of PROM designs a Hex list is printed out.

MARC and gate circuits are documented by Test and Program Listings.

## Computer Hardware

LOG/iC can be run on a number of computers (16 or 32-bit) with different operating systems.

Porting to other hardware is possible, especially if the operating system is already supported. ISDATA will check the portability on request.

| COMPUTER | OPERATING SYSTEM | STORAGE MEDIUM | HARDWARE |
|---|---|---|---|
| VAX | VMS (V. 4.0)<br>ULTRIX (V 1.2) | MAGNETIC TAPE (1600 BPI)<br>CARTRIDGE TK 50 | VT 100 TERMINALS OR COMPAT. |
| MICROVAX | MICROVMS | | |
| APOLLO | AEGIS | 5 1/4" DISKETTE | |
| HP 9000 | HP–UX | 1/4" TAPE CARTRIDGE | MOD. 200<br>MOD. 300 |
| IBM PC/XT*<br>IBM PC/AT* | MS–DOS (MIN. PC–DOS V. 2.0) | 5 1/4" DISKETTE (360 KB) | |

418 11

*or compatible

## Integration into the CAE Environment

LOG/iC generates the programming data for PLDs in standard data formats, such as JEDEC and Intel-Hex. The connection to the programmer is supported from the screen by means of a special communication module, which emulates various hardware and software handshakes. All interface parameters are software selectable, so that LOG/iC can be run together with any programmer.

Optional interfaces make possible the input of net lists and circuit schematics. LOG/iC is therefore compatible with third-party systems and can be integrated into an existing environment. This is also guaranteed on the output level by the post-processors of the module package "Gates". These post-processors output the gate structure of the design in the form of net lists in various common data formats, so that it is possible to transfer the data produced by LOG/iC to third-party systems.

## Support of the Designer

The user of LOG/iC is supported in many ways. With every installation comes a handbook that gives extensive information on the LOG/iC modules and their operation. The handbooks are delivered in English or German. A great variety of examples show the range of options offered by the LOG/iC syntax and make it easier for the user to get familiar with it.

Function "Help" of the menu gives answers to specific questions. Depending on the kind of installation, help is offered in English or German. The user can install the software. Explicit instructions are given in the handbook and in the installation program.

A maintenance contract assures you of prompt updates for the program, that also support new devices. In addition, it guarantees you immediate help through ISDATA or its representatives in case of any application problems.

## Ordering Information

Each delivery contains the handbook and the storage medium for the respective CPU. You can find the order numbers in the respective data sheets.

## Example of PAL32VX10 Support

The following example demonstrates the ability to use the T-type flip-flop in the PAL32VX10 directly though LOG/iC software. LOG/iC automatically emulates the T-type flip-flop as needed through specific programming of the XOR terms in the PAL32VX10. The following file fully describes a modulus-93 counter.

```
*IDENTIFICATION
 BIT STREAM COUNTER FOR 93 BITS (VER.2.1_T-FLIPFLOP)
 GUENTER BIEHL
 ISDATA KARLSRUHE, TEL. 0721 693092
*DECLARATIONS
 X-VARIABLES = 3
 Y-VARIABLES = 1
 Z-VARIABLES = 7
*X-NAMES
 RESET = 1, DOWN=2, COUNT=3;
*Y-NAMES
 CARRY = 1;
*Z-NAMES
 QQ[6..0] = [7..1];
*RUN-CONTROL
 LISTING = PINOUT, FUSE-PLOT, EQUATIONS;
 PROGFORMAT =JEDEC;
*Z-VALUES
 S[1..93] = [0..92];
*FLOW-TABLE
;COUNTER WITH 93 STATES AND CARRY SIGNAL
 S[1..93], X1--, Y0, F1;          RESET CONDITION
 S[1..93], X00-, Y0, F[1..93];    HOLD
 S[1..92], X010, Y0, F[2..93];    COUNT UP
 S93 ,     X010, Y1, F1;          CARRY
 S[2..93], X011, Y0, F[1..92];    COUNT DOWN
 S1 ,      X011, Y1, F93;         CARRY
*STATE-ASSIGNMENT
 Z-VALUES;
*PAL
 TYPE=PAL32VX10;
*PINS
 RESET=3, COUNT=4, DOWN=5,
 CARRY=22, QQ[0..6]=[15..21];
*FLI
 T-FLIPFLOP
*END
```

The product terms are minimized by LOG/iC for inputs to the T-type flip-flops. Compared to a D-type flip-flop implementation, only about half as many product terms are required.

The Boolean equations are generated automatically by LOG/iC software. In the equations for the T-type flip-flops, the output signals are distinguished by the suffix "T". In addition to the equations, LOG/iC generates information on the resulting configuration of the macrocells and the XOR terms.

```
BIT STREAM COUNTER FOR 93 BITS (VER.2.1_T-FLIPFLOP)
GUENTER BIEHL
ISDATA KARLSRUHE, TEL. 0721 693092
28-OCT-86 12:29:35

*****************************************************************
***                      BOOLEAN EQUATIONS                    ***
*****************************************************************

QQ0:T    := /QQ1  & /QQ2  & /QQ3  & /QQ4  & /QQ5  & /QQ6
                  & /RESET&  DOWN &  COUNT
          +  QQ1  &  QQ2  &  QQ3  &  QQ4  &  QQ5  &  QQ6
                  & /RESET&  DOWN & /COUNT
          +  QQ0  &  RESET
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  &  DOWN & /COUNT ;

QQ1:T    :=  QQ2  &  QQ3  &  QQ4  &  QQ5  &  QQ6  & /RESET
                  &  DOWN & /COUNT
          +  QQ1  &  RESET
          +  QQ1  & /QQ2  & /QQ3  & /QQ4  & /QQ5  & /QQ6
                  &  DOWN &  COUNT
          +  QQ0  & /QQ2  & /QQ3  & /QQ4  & /QQ5  & /QQ6
                  & /RESET&  DOWN &  COUNT ;

QQ2:T    := /QQ3  & /QQ4  & /QQ5  & /QQ6  & /RESET& DOWN
                  &  COUNT
          +  QQ3  &  QQ4  &  QQ5  &  QQ6  & /RESET& DOWN
                  & /COUNT
          +  QQ2  &  RESET
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  &  DOWN & /COUNT ;

QQ3:T    := /QQ4  & /QQ5  & /QQ6  & /RESET& DOWN  &  COUNT
          +  QQ4  &  QQ5  &  QQ6  & /RESET& DOWN  & /COUNT
          +  QQ3  &  RESET
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  & DOWN  & /COUNT ;

QQ4:T    := /QQ5  & /QQ6  & /RESET&  DOWN &  COUNT
          +  QQ5  &  QQ6  & /RESET&  DOWN & /COUNT
          +  QQ4  &  RESET
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  &  DOWN   ;

/QQ5:T   := /RESET& /DOWN
          + /QQ6  & /RESET& /COUNT
          +  QQ6  & /RESET&  COUNT
          + /QQ5  &  RESET
          + /QQ0  & /QQ1  & /QQ2  & /QQ3  & /QQ4  & /QQ5
                  &  COUNT ;

/QQ6:T   := /QQ0  & /QQ1  & /QQ2  & /QQ3  & /QQ4  & /QQ5
                  & /QQ6  &  COUNT
          + /RESET& /DOWN
          + /QQ6  &  RESET
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  & /COUNT ;
CARRY    := /QQ0  & /QQ1  & /QQ2  & /QQ3  & /QQ4  & /QQ5
                  & /QQ6  & /RESET&  DOWN &  COUNT
          +  QQ0  &  QQ2  &  QQ3  &  QQ4  & /RESET&  DOWN
                  & /COUNT ;
```

ISDATA GmbH
Haid-und-Neu Strasse 7
D-7500 Karlsruhe 1
West Germany
Tel: 0721/693092

## Representatives

**Switzerland**

CAS
Computer Access Systems AG
Papiermuehlestr. 145
CH-3063 Ittigen
Tel.: 031-587844

**Italy**

Instrumatic
Via Piave u. 22/A
I-20016 Pero (Milano)
Tel.: 02-3538041

**Spain**

Instrumatic Espanola SA
Paseo de la Castellana 127-2-A
E-28046 Madrid
Tel.: 01-455 8112

**Great Britain**

Instrumatic U.K. Ltd.
First Avenue
Globe Park
Marlow, Bucks. SL7 1YA
Tel.: 06284-76741

**Netherlands**

Diode Nederland
Meidoornkade 22
NL-3992 AE Houten
Tel.: 03403-91234

**Scandinavia**

Pronesto AB
Saab-Scania Combitech Group
BOX 1358
S-171 26 Solna
Sweden
Tel.: 08-733 93 00

**France**

AK Division Electronique
54 Avenue Emile-Zola
F-75015 Paris
France
Tel.: 0033-1-45 75 53 53

## OEM Partners

Kontron
Kontron Metechnik GmbH
Oskar von Miller Str. 17
D-8057 Eching
Tel.: 08165-77-1

Kontron
630 Clyde Ave.
Mountain View, CA 94039-7230
(415) 965-7020

Elan Digital Systems, Ltd./U.K.
16 -20 Kelvin Way
Crawley, West Sussex
RH10 2TS
Tel.: 0293-510448

Digelec AG
Doerfli Strasse 14
CH-8057 Zurich
Tel.: 01-312 4622

Digelec
1602 Lawrence Ave.
Suite 113
Ocean, NJ 07712
(800) 367-8750

SMS
Microcomputer-Systeme
Im Morgental 13
D-8994 Hergatz
Tel.: 07522-4460

Micropross
Parc d'activite des Pres
5, rue Denis-Papin
59650 Villeneuve d'Ascq
France
Tel.: 20 47 90 40

# Programming

There are two common situations when a PAL device user wants to program parts:

1. The user has a master device and wants to program the master pattern into new unprogrammed parts from the same or from a different manufacturer.

2. The user has a file that is in JEDEC standard Programmable Logic Data Transfer Format and wants to send the file to a programmer and program parts to that pattern.

All approved programmers can accomplish either of these tasks. You will have to refer to your programmer manual for detailed procedures, but here are some general guidelines:

## Programming with the Use of a Master Device

Suppose you have a master device and you want to program a device of the same type with exactly the same pattern. The master device can be an MMI or AMD device or another manufacturer's functionally equivalent device. Follow these steps:

1. Set the programmer to read (or copy) the master device. This may require having a hardware adaptor for the master and entering a product code unique to the manufacturer and device type.

2. Install the correct adaptor (if required). Enter the appropriate product code information or select the device type from the menu. Then place the master device in the correct socket and read its fuse pattern into the programmer memory. Use whatever operating sequence is required by the programmer for this operation.

3. The pattern is now in the programmer memory and will remain there until the memory is cleared or the programmer power is turned off. Changing an adaptor or product code will not erase the memory. Usually at the end of a copy operation a checksum will be displayed. Make a note of this number. The checksum is a calculated hexadecimal code for the pattern loaded into memory. It can be very helpful in diagnosing any programming problems. If a part is to be re-used frequently as a master device it is a good practice to write the checksum on the top of the part. Never proceed with programming without checksum agreement after reading a master.

### Error Detection

As a matter of curiosity take the part out of the socket once and read an empty socket. Also read a known blank part (using the right adaptor). Checksums from these two situations will be helpful in diagnosing two common problems when programming from masters:

- Forgetting to lock down the socket lever to make good contact after loading a part
- Loading an unprogrammed part as a master by mistake.

4. Now prepare the programmer for the device to be programmed with the master pattern loaded into memory. Some programmers require different adaptors for different manufacturer's parts. If the programmer being used has this requirement, be sure to use the proper adaptor for the exact part number to be programmed. Using the wrong adaptor can cause permanent damage to the parts. Always check for adaptor compatibility.

5. Everything's OK. You have the correct adaptor, the right device code (or have selected the device from the menu) and you wrote down the checksum that you got after loading the master. Now put the programmer in the mode used for programming from its memory and execute the programming operation.

There is some variation in the sequence of events carried out by different programmers during the programming cycle, but all of them program and verify the appropriate fuses to match the pattern in the programmer memory. Such operations as Blank Checks, Illegal Bit Checks, Test Vector Testing, and Security Fuse Programming can be a part of the programming sequence. Check the programmer manufacturer's manual for the availability and appropriate use of these features.

The essential part of the programming cycle is the programming and verification of each fuse followed by a verification of all fuses at both low and high Vcc. At the very end of the programming sequence you will see the checksum for the part you have just programmed. This checksum should agree with the master part checksum. You now have a programmed part that is functionally identical to the master.

## Programming From a JEDEC File

A JEDEC standard file is the output of design software packages used to specify programming pattern information to a programmer. All approved programmers will accept JEDEC files. A JEDEC file is normally generated on a computer by PLD design software. The unique aspect of programming from a JEDEC file is the transfer of the file to the programmer. After the file has been transferred into the programmer, the programming task is identical to programming from a master with one exception. The exception is that design software may be used to prepare test vectors to be applied to a device immediately following the programming cycle. These vectors will be transmitted with the JEDEC fuse file and they have a JEDEC standard format of their own.

General guidelines for transfer of a JEDEC file and programming are as follows:

1. Make sure your file is in the standard JEDEC format. This will not be a problem if you are using software for file preparation that adheres to this standard.

2. Connect the JEDEC file source to the programmer with an RS232 cable. The programmer manual will describe the connection details.

3. Prepare the programmer for receiving a JEDEC file over a link. This will generally involve entering the product code information and putting the programmer in a ready-to-receive mode.

4. Transmit the file from the computer source using commercially available communications software or operating system commands.

5. After transmission a checksum should appear on the programmer display. Part of the JEDEC standard file is a checksum. If the displayed checksum is the same as the JEDEC file generated checksum transmission has been successful.

6. Program a PAL device by first installing the correct adaptor (if needed) and then entering the programming mode. Finally put a part in the socket and execute the programming operation.

## Register Preload

Register preload is an aid to functional testing of registered PAL devices. Functional testing is usually performed after a device is programmed but before it is installed on the circuit board. Functional testing exercises the functional logic circuitry of a device that is not fully testable prior to programming, providing a higher final quality level for programmable products. For a more thorough discussion of functional testing and related quality issues see the "ProPAL/HAL/ZHAL" section, page 3-104.

Using register preload, the registers of a device can be "preloaded" to any desired state value. The ability to set the registers to any arbitrary value is extremely useful for testing state machine designs where the output is fed back into the array as an input. It lets the user check for deadlock loops and proper recovery from illegal states. It also simplifies testing state transitions of states which may be difficult to reach through normal state transitions.

Consider the example of the 6-state counter illustrated in the state machine diagram of Figure 1.

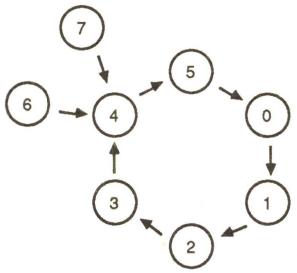



**Figure 1.** 419 1

States 6 and 7 are illegal states, both transitioning to state 4. If the registers of the device are not preloadable, it is difficult to check for recovery from these states since they cannot be reached through normal state transitions. If register preload is available, however, it is a simple matter to write a set of vectors that sets the device to the illegal state and then clocks it and checks proper recovery.

If the device powers up in state 1 and you want to test the transition from state 0 → 1, the only way to check that transition is to write a series of vectors that cycle the device through the state sequence starting at state 1, until the desired state, state 0, is reached. With register preload, reaching state 0 is simply a matter of writing a vector that sets the device to that state.

In general, register preload simplifies the task of writing test vectors for functional testing, and is especially helpful in testing the conditions described in this discussion. However, test vectors written utilizing register preload can provide only a limited amount of functional coverage. Full coverage can only be achieved when the vectors used to test the device simulate actual operating conditions, and preload is not a normal operating condition.

### Programmer Support

Not all programmers support register preload. The programmer guide lists the programmers that do support this feature. For more specific information regarding your programmer, contact the manufacturer.

## Choosing the Right Programmer

Monolithic Memories has qualified several PAL device programmers, and choosing among them is not simple. You must consider many factors. Does the programmer handle all of the devices you will be using? Does it program PAL devices, sequencers, PLE devices, and PROMs? Does it program TTL, EPROM CMOS, EEPROM CMOS, and ECL technology products? How easily is it upgraded for future devices? Does it have provisions for test vectors, accepting JEDEC files, or a handler interface? And what about cost?

APPROVED PROGRAMMERS—PRODUCTS & FEATURES—CHART

| PROGRAMMER | DATA I/O | | | STAG | | DIG-ELEC | KONT- | STORY | SDI | VARIX | MICRO-PROSS | JMC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 29 | M60 | US40 | PPZ | ZL30 | 803 | RON | SYS. | 1000 | OMNI | 5000 | P3 |
| **FEATURES** | | | | | | | | | | | | |
| PRICE | M | M | H | M | L | M | M | L | L | M | M | L |
| TEST VECTORS | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| PRELOAD | Y | Y | Y | N | N | N | N | Y | N | Y | Y | Y |
| ACCEPTS JEDEC | Y | Y | Y | Y | Y | Y | Y | Y | N | Y | Y | Y |
| PIN CONTINUITY | Y | Y | Y | Y | Y | N | Y | N | Y | N | N | N |
| FINGERPRINT | Y | Y | N | N | N | N | N | N | N | N | N | N |
| HANDLER | Y | Y | Y | Y | Y | N | N | N | Y | N | N | N |
| CRT | N | N | N | Y | N | Y | N | N | N | N | Y | N |
| STANDALONE | Y | Y | N | Y | Y | Y | Y | Y | N | N | Y | Y |
| SECURITY CHECK | Y | Y | Y | Y | N | N | Y | N | N | N | N | N |
| FAST PROGRAM | N | Y | Y | Y | Y | N | N | N | N | N | N | N |
| ASSEMBLER | Y | N | N | Y | N | Y | N | N | Y | N | N | N |
| DISASSEMBLER | N | N | N | N | N | Y | N | N | N | N | Y | N |
| 1 PULSE PROG. | Y | Y | Y | N | N | N | N | N | N | N | N | N |

NOTE: Y=YES N=NO

Despite these variations in features, today's programmers fall into two broad categories, PC-based programmers and standalone programmers. PC-based programmers consist of a board (that plugs into a PC) and an external box with socket(s) for the device being programmed. The plug-in board contains the "intelligence" of the programmer, and, as the name implies, these programmers require a PC for use. Standalone programmers can perform all programming operations without a PC. PC-based programmers are usually lower cost and lower performance design and development tools that support a limited number of devices and have limited capabilities. Standalone programmers offer higher performance (e.g., faster programming), and are oriented to the production environment.

The chart above lists programmer features, as well as current support for some of the newer part types. This information should help you decide on the best programmer for your needs.

## Approved Programmers

Monolithic Memories and Advanced Micro Devices PAL devices are manufactured under strict processing procedures to provide our PAL device users with the highest-quality PAL devices available. We take the same approach in our programmer vendor approval process, and recommend that you choose an approved programmer for your PAL device programming needs.

### The Benefits of Using Approved Programmers

When you choose an approved programmer you gain all the benefits of our thorough vendor evaluation. You can feel confident investing in equipment that will give you consistently reliable results and will be able to support current and future generations of programmable logic devices. When you select an approved programmer you get many benefits:

- Your programmed product is backed by our corporate warranty.
- New features and algorithm updates are quickly implemented.
- Any new programmer software and hardware releases are factory evaluated and approved before release.
- Your equipment will have a long "technical lifetime".
- Through our sales force and Field Applications Engineers you have a factory interface with the programmer vendors to deal with any issues or concerns that might arise.

### The Approval Process

The Programmability Group at Monolithic Memories works closely with our programmer vendors to ensure that high-quality programming and testing support is available to all users of our PLDs.

To gain approval a programmer must pass a rigorous series of tests which include:

- Conformity to programming specifications.
- Devices programmed must pass all reliability tests. This reliability testing is performed by Monolithic Memories as part of the evaluation.
- Programmer must meet programming yield requirements for both array and security fuse programming.
- Programmer must be able to support JEDEC format files and communication standards.

Programmer vendors are encouraged to support structured test vector testing and preload capability, pin continuity and pre-programming security fuse checking.

### New Product Support

Approved programmers must also provide timely support for new products and programming algorithm updates and revisions. This ensures PAL device users that no matter which approved programmer they choose, they can feel confident that it will support the latest and greatest PLDs we have to offer.

Additionally, we work closely with our approved vendors in the development of their new programmers and our new PLDs. This means that their new products will be able to support our future PLD offerings.

### A Broad Range of Programmers

We work with a broad range of programmer vendors, so you can find a programmer to suit your engineering needs as well as your budget. Approved programmers cover the range from economical engineering/design prototyping tools to high-volume production units. We have a worldwide programmer vendor base, so programming support is available no matter where you use PAL devices. Approved programmers are available from American, British, French, German and Japanese vendors.

Our quality and reliability guarantees are made for products programmed with approved programmers only. Use of unapproved programmers voids our corporate warranty and may result in poor manufacturing yields and product performance. The Approved Programmer section in this handbook is a valuable tool. The information it contains can help you obtain consistent, reliable high-quality programming results with your PLDs.


IMPORTANT
NOTES

# Programmer Reference Guide
## Table of Contents


Adams MacDonald Enterprises, Inc. ..... 3-82

Data I/O ..... 3-84

Digelec ..... 3-88

Kontron ..... 3-90

Logical Devices, Inc. ..... 3-92

Microprоss ..... 3-94

Stag Microsystems ..... 3-96

Storey Systems ..... 3-98

Structured Design ..... 3-100

Varix ..... 3-102

## Adams MacDonald Enterprises, Inc.

1.0 PROMAC P3

(408) 373-3607

20 Pin Device Families

| Family | Product | Software Rev. | S1/S2 |
|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — |
| Asynchronous | PAL16RA8 | 3.0 | 1/12 |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 3.0<br>3.0<br>3.0<br>3.0 | 1/0<br>1/1<br>1/2<br>1/3 |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | 3.0<br>3.0<br>3.0<br>3.0 | 5/0<br>5/1<br>5/2<br>5/3 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | 3.0<br>3.0<br>3.0<br>3.0 | 5/0<br>5/1<br>5/2<br>5/3 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | 3.0<br>3.0<br>3.0<br>3.0 | —<br>—<br>—<br>— |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 3.0<br>3.0<br>3.0<br>3.0 | 0/10<br>0/11<br>0/12<br>0/13 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | 3.0<br>3.0<br>3.0 | —<br>—<br>— |
| Arithmetic | PAL16X4 | 3.0 | 0/14 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | 3.0 | — |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 3.0<br>3.0<br>3.0<br>3.0<br>3.0<br>3.0<br>3.0<br>3.0<br>3.0 | 0/1<br>0/6<br>0/2<br>0/7<br>0/3<br>0/8<br>0/4<br>0/9<br>0/5 |

24 Pin and MegaPAL Device Families

| Family | Product | Software Rev. | S1/S2 |
|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — |
| Varied with XOR | PAL32VX10/A | — | — |
| Varied Product Terms | AmPAL22V10/-15/A | 3.0 | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | — |
| Registered XOR | PAL22RX8A | — | — |

## Adams MacDonald Enterprises, Inc.

| Family | Product | Software Rev. | S1/S2 |
|---|---|---|---|
| **Asynchronous** | PAL20RA10-20 | 3.00 | 3/4 |
| | PAL20RA10 | 3.00 | 3/4 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | — |
| **PAL20RS10 Shared Product Terms** | PAL20S10 | 3.0 | 3/5 |
| | PAL20RS10 | 3.0 | 3/6 |
| | PAL20RS8 | 3.0 | 3/7 |
| | PAL20RS4 | 3.0 | 3/8 |
| **PAL20X10A Exclusive OR** | PAL20L10A | 3.0 | 2/7 |
| | PAL20X10A | 3.0 | 2/15 |
| | PAL20X8A | 3.0 | 3/0 |
| | PAL20X4A | 3.0 | 3/1 |
| **PAL20X10 Exclusive OR** | PAL20L10 | 3.0 | 2/7 |
| | PAL20X10 | 3.0 | 2/12 |
| | PAL20X8 | 3.0 | 3/13 |
| | PAL20X4 | 3.0 | 3/14 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A | — | — |
| | AmPAL20RP10B/AL/A | — | — |
| | AmPAL20RP8B/AL/A | — | — |
| | AmPAL20RP6B/AL/A | — | — |
| | AmPAL20RP4B/AL/A | — | — |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2 | 3.0 | 2/8 |
| | PAL20R8B/B-2/A/A-2 | 3.0 | 2/9 |
| | PAL20R6B/B-2/A/A-2 | 3.0 | 2/10 |
| | PAL20R4B/B-2/A/A-2 | 3.0 | 2/11 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45 | — | — |
| | PALC20R8Z-35/-45 | — | — |
| | PALC20R6Z-35/-45 | — | — |
| | PALC20R4Z-35/-45 | — | — |
| **Decoder** | PAL6L16A | 3.0 | 3/11 |
| | PAL8L14A | 3.0 | 3/10 |
| **PAL12L10 Combinatorial** | PAL12L10 | 3.0 | 2/2 |
| | PAL14L8 | 3.0 | 2/3 |
| | PAL16L6 | 3.0 | 2/4 |
| | PAL18L4 | 3.0 | 2/5 |
| | PAL20L2 | 3.0 | 2/6 |
| | PAL20C1 | 3.0 | 2/1 |
| **MegaPAL Device** | PAL32R16 | — | — |
| **PROSE Device** | PMS14R21/A | — | — |
| **Programmable Logic Sequencer** | PLS105-37 | — | — |
| | PLS167-33 | — | — |
| | PLS168-33 | — | — |
| **Fuse Programmable Controller** | Am29PL141 | — | — |
| **Programmable Event Generator** | Am2971 | — | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — | — |
| **ECL Combinatorial ECL Latched** | PAL10H20P8 | — | — |
| | PAL10H20G8 | — | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Data I/O

(800) 247-5700

1.1 Logic Pak 303A
1.1.1 P/T Adapter 303A-002
1.1.2 303A-ECL
1.1.3 303A-011A/B
2.0 System 19, 29A, 29B, Unisite 40
3.0 System 19, 22, 29A, 29B, Unisite 40

### 20 Pin Device Families

| Family | Product | Adapter | Generic Adapter | Code |
|---|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — | 97-84 |
| Asynchronous | PAL16RA8 | 303A-002-V08 | 303A-011A/B-V01 | 22-30 |
| PAL16RP8A Programmable Polarity | PAL16P8A | 303A-002-V08 | 303A-011A/B-V01 | 22-30 |
| | PAL16RP8A | 303A-002-V08 | 303A-011A/B-V01 | 22-31 |
| | PAL16RP6A | 303A-002-V08 | 303A-011A/B-V01 | 22-31 |
| | PAL16RP4A | 303A-002-V08 | 303A-011A/B-V01 | 22-31 |
| PAL16R8-10 | PAL16L8-10/H-15 | — | 303A-011A/B-V02 | 22-17 |
| | PAL16R8-10/H-15 | — | 303A-011A/B-V02 | 22-67 |
| | PAL16R6-10/H-15 | — | 303A-011A/B-V02 | 22-67 |
| | PAL16R4-10/H-15 | — | 303A-011A/B-V02 | 22-67 |
| PAL16R8D/B | PAL16L8D/B | 303A-002-V08 | 303A-011A/B-V01 | 30-17 |
| | PAL16R8D/B | 303A-002-V08 | 303A-011A/B-V01 | 30-24 |
| | PAL16R6D/B | 303A-002-V08 | 303A-011A/B-V01 | 30-24 |
| | PAL16R4D/B | 303A-002-V08 | 303A-011A/B-V01 | 30-24 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L | — | 303A-011A/B-V05 | 97-81 |
| | AmPAL16R8/B/AL/A/Q/L | — | 303A-011A/B-V05 | 97-80 |
| | AmPAL16R6/B/AL/A/Q/L | — | 303A-011A/B-V05 | 97-82 |
| | AmPAL16R4/B/AL/A/Q/L | — | 303A-011A/B-V05 | 97-17 |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4 | 303A-002-V08 | 303-011A/B-V01 | 22-17 |
| | PAL16R8/B-2/B-4/A/A-2/A-4 | 303A-002-V08 | 303-011A/B-V01 | 22-24 |
| | PAL16R6/B-2/B-4/A/A-2/A-4 | 303A-002-V08 | 303-011A/B-V01 | 22-24 |
| | PAL16R4/B-2/B-4/A/A-2/A-4 | 303A-002-V08 | 303-011A/B-V01 | 22-24 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25 | — | 303A-011A/B-V04 | DB-17 |
| | PALC16R8Q-25 | — | 303A-011A/B-V04 | DB-24 |
| | PALC16R6Q-25 | — | 303A-011A/B-V04 | DB-24 |
| | PALC16R4Q-25 | — | 303A-011A/B-V04 | DB-24 |
| AmPAL16HD8 | AmPAL16H8A/L | — | 303-011A/B-V05 | 97-25 |
| | AmPAL16HD8A/L | — | 303-011A/B-V05 | 97-25 |
| | AmPAL16LD8A/L | — | 303-011A/B-V05 | 97-17 |
| Arithmetic | PAL16X4 | 303A-002-V08 | 303A-011A/B-V01 | 22-24 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — | 303A-011A/B-V01 | 97-29 |
| PAL10H8 Combinatorial | PAL10H8/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-18 |
| | PAL10L8/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-13 |
| | PAL12H6/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-19 |
| | PAL12L6/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-14 |
| | PAL14H4/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-20 |
| | PAL14L4/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-15 |
| | PAL16H2/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-22 |
| | PAL16L2/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-16 |
| | PAL16C1/-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-21 |

### 24 Pin and MegaPAL Device Families

| Family | Product | Adapter | Generic Adapter | Code |
|---|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — | — |
| Varied with XOR | PAL32VX10/A | — | 303A-011A/B-V01 | 22-77 |
| Varied Product Terms | AmPAL22V10/-15/A | — | 303A-011A/B-V01 | 97-28/83 |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | 303A-011A/B-V02 | DB-28 |
| Registered XOR | PAL22RX8A | — | 303A-011A/B-V01 | 22-78 |

## Data I/O

### 20 Pin Device Families

| Family | Product | M60 Rev. | Unisite 40 Rev. | System | Pak |
|---|---|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | V11 | — | ALL | 303A-V04 |
| Asynchronous | PAL16RA8 | V05 | 1.54 | ALL | 303A-V04 |
| PAL16RP8A Programmable Polarity | PAL16P8A | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16RP8A | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16RP6A | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16RP4A | V05 | 1.54 | ALL | 303A-V04 |
| PAL16R8-10 | PAL16L8-10/H-15 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R8-10/H-15 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R6-10/H-15 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R4-10/H-15 | V05 | 1.54 | ALL | 303A-V04 |
| PAL16R8D/B | PAL16L8D/B | V11.1 | 1.54 | ALL | 303A-V04 |
| | PAL16R8D/B | V11.1 | 1.54 | ALL | 303A-V04 |
| | PAL16R6D/B | V11.1 | 1.54 | ALL | 303A-V04 |
| | PAL16R4D/B | V11.1 | 1.54 | ALL | 303A-V04 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L | V11.1 | 1.3 | ALL | 303A-V04 |
| | AmPAL16R8/B/AL/A/Q/L | V11.1 | 1.3 | ALL | 303A-V04 |
| | AmPAL16R6/B/AL/A/Q/L | V11.1 | 1.3 | ALL | 303A-V04 |
| | AmPAL16R4/B/AL/A/Q/L | V11.1 | 1.3 | ALL | 303A-V04 |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R8/B-2/B-4/A/A-2/A-4 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R6/B-2/B-4/A/A-2/A-4 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16R4/B-2/B-4/A/A-2/A-4 | V05 | 1.54 | ALL | 303A-V04 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25 | — | — | ALL | 303A-V04 |
| | PALC16R8Q-25 | — | — | ALL | 303A-V04 |
| | PALC16R6Q-25 | — | — | ALL | 303A-V04 |
| | PALC16R4Q-25 | — | — | ALL | 303A-V04 |
| AmPAL16HD8 | AmPAL16H8A/L | V03 | 1.3 | ALL | 303A-V04 |
| | AmPAL16HD8A/L | V03 | 1.3 | ALL | 303A-V04 |
| | AmPAL16LD8A/L | V03 | 1.3 | ALL | 303A-V04 |
| Arithmetic | PAL16X4 | V05 | 1.54 | ALL | 303A-V04 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | V05 | 1.3 | ALL | 303A-V04 |
| PAL10H8 Combinatorial | PAL10H8/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL10L8/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL12H6/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL12L6/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL14H4/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL14L4/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16H2/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16L2/-2 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL16C1/-2 | V05 | 1.54 | ALL | 303A-V04 |

### 24 Pin and MegaPAL Device Families

| Family | Product | M60 Rev. | Rev. | Unisite 40 System | Pak |
|---|---|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — | — | — |
| Varied with XOR | PAL32VX10/A | V10 | 1.54 | ALL | 303A-V04 |
| Varied Product Terms | AmPAL22V10/-15/A | V03 | 1.3 | ALL | 303A-V04 |
| Varied Terms (CMOS) | PALC22V10H-25/35 | V10 | 1.54 | ALL | 303A-V04 |
| Registered XOR | PAL22RX8A | V10 | 1.54 | ALL | 303A-V04 |

## Data I/O

### 24 Pin and MegaPAL™ Device Families

| Family | Product | Adapter | Generic Adapter | Code |
|---|---|---|---|---|
| Asynchronous | PAL20RA10-20 | — | — | — |
| | PAL20RA10 | 303A-002-V08 | 303A-011A/B-V01 | 22-45 |
| AmPAL20XRP10 | AmPAL22XP10-20/-30L/-30/-40L | — | 303A-011A/B-V02 | 97-2C |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | 303A-011A/B-V02 | 97-0E |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | 303A-011A/B-V02 | 97-0D |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | 303A-011A/B-V02 | 97-0C |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | 303A-011A/B-V02 | 97-0B |
| PAL20RS10 Shared Product Terms | PAL20S10 | 303A-002-V08 | 303A-011A/B-V01 | 22-43 |
| | PAL20RS10 | 303A-002-V08 | 303A-011A/B-V01 | 22-44 |
| | PAL20RS8 | 303A-002-V08 | 303A-011A/B-V01 | 22-44 |
| | PAL20RS4 | 303A-002-V08 | 303A-011A/B-V01 | 22-46 |
| PAL20X10A Exclusive OR | PAL20L10A | 303A-002-V08 | 303A-011A/B-V01 | 22-06 |
| | PAL20X10A | 303A-002-V08 | 303A-011A/B-V01 | 22-36 |
| | PAL20X8A | 303A-002-V08 | 303A-011A/B-V01 | 22-36 |
| | PAL20X4A | 303A-002-V08 | 303A-011A/B-V01 | 22-36 |
| PAL20X10 Exclusive OR | PAL20L10 | 303A-002-V08 | 303A-011A/B-V01 | 22-06 |
| | PAL20X10 | 303A-002-V08 | 303A-011A/B-V01 | 22-23 |
| | PAL20X8 | 303A-002-V08 | 303A-011A/B-V01 | 22-23 |
| | PAL20X4 | 303A-002-V08 | 303A-011A/B-V01 | 22-23 |
| AmPAL20L10 | AmPAL20L10B/-20/AL | — | 303A-011A-V02 | 97-06 |
| AmPAL20RP10 | AmPAL22P10B/AL/A | — | 303A-011A-V02 | 97-2B |
| | AmPAL20RP10B/AL/A | — | 303A-011A-V02 | 97-9F |
| | AmPAL20RP8B/AL/A | — | 303A-011A-V02 | 97-9E |
| | AmPAL20RP6B/AL/A | — | 303A-011A-V02 | 97-9D |
| | AmPAL20RP4B/AL/A | — | 303A-011A-V02 | 97-9C |
| PAL20R8B/B-2*/A/A-2 | PAL20L8B/B-2/A/A-2 | 303A-002-V08 | 303A-011A/B-V01 | 22-26 |
| | PAL20R8B/B-2/A/A-2* | 303A-002-V08 | 303A-011A/B-V01 | 22-27 |
| | PAL20R6B/B-2/A/A-2* | 303A-002-V08 | 303A-011A/B-V01 | 22-27 |
| | PAL20R4B/B-2/A/A-2* | 303A-002-V08 | 303A-011A/B-V01 | 22-27 |
| PALC20R8Z Zero Standby Power | PALC20L8Z-35/-45 | 303A-002-V08 | 303A-011A/B-V02 | 46-26 |
| | PALC20R8Z-35/-45 | 303A-002-V08 | 303A-011A/B-V02 | 46-27 |
| | PALC20R6Z-35/-45 | 303A-002-V08 | 303A-011A/B-V02 | 46-27 |
| | PALC20R4Z-35/-45 | 303A-002-V08 | 303A-011A/B-V02 | 46-27 |
| Decoder | PAL6L16A | 303A-002-V08 | 303A-011A/B-V01 | 22-48 |
| | PAL8L14A | 303A-002-V08 | 303A-011A/B-V01 | 22-49 |
| PAL12L10 Combinatorial | PAL12L10 | 303A-002-V08 | 303A-011A/B-V01 | 22-01 |
| | PAL14L8 | 303A-002-V08 | 303A-011A/B-V01 | 22-02 |
| | PAL16L6 | 303A-002-V08 | 303A-011A/B-V01 | 22-03 |
| | PAL18L4 | 303A-002-V08 | 303A-011A/B-V01 | 22-04 |
| | PAL20L2 | 303A-002-V08 | 303A-011A/B-V01 | 22-05 |
| | PAL20C1 | 303A-002-V08 | 303A-011A/B-V01 | 22-12 |
| MegaPAL Device | PAL32R16 | 303A/008-V02 | — | 22-47 |
| PROSE Device | PMS14R21/A | — | 303A-011A/B-V02 | 22-58 |
| Programmable Logic Sequencer | PLS105-37 | — | 303A-011A/B-V03 | 2A-63 |
| | PLS167-33 | — | 303A-011A/B-V03 | 2A-60 |
| | PLS168-33 | — | 303A-011A/B-V03 | 2A-74 |
| Fuse Programmable Controller | Am29PL141 | — | 303A-FPC-V01 | 97-79 |
| Programmable Event Generator | Am2971 | — | 303A-011A-V03 | — |
| ECL Registered | PAL10H/10020EV/EG8 | 303A-ECL | — | — |
| ECL Combinatorial | PAL10H20P8 | 303A-ECL | — | 22-42 |
| ECL Latched | PAL10H20G8 | 303A-ECL | — | 22-42 |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.
*PAL 20R8 B-2 family code using preload is 22-68 with Generic Adapter V04.

## Data I/O

### 24 Pin and MegaPAL Device Families

| Family | Product | M60 Rev. | Unisite 40 Rev. | System | Pak |
|---|---|---|---|---|---|
| Asynchronous | PAL20RA10-20 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL20RA10 | V05 | 1.54 | ALL | 303A |
| AmPAL20XRP10 | AmPAL22XP10-20/-30L/-30/-40L | — | — | ALL | 303A-V04 |
| | AmPAL20XRP10-20/-30L/-30/-40 | V10 | — | ALL | 303A-V04 |
| | AmPAL20XRP8-20/-30L/-30/-40L | V10 | — | ALL | 303A-V04 |
| | AmPAL20XRP6-20/-30L/-30/-40L | V10 | — | ALL | 303A-V04 |
| | AmPAL20XRP4-20/-30L/-30/-40L | V10 | — | ALL | 303A-V04 |
| PAL20RS10 Shared Product Terms | PAL20S10 | V05 | 1.54 | ALL | 303A |
| | PAL20RS10 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL20RS8 | V05 | 1.54 | ALL | 303A-V04 |
| | PAL20RS4 | V05 | 1.54 | ALL | 303A-V04 |
| PAL20X10A Exclusive OR | PAL20L10A | V05 | 1.54 | ALL | 303A |
| | PAL20X10A | V05 | 1.54 | ALL | 303A |
| | PAL20X8A | V05 | 1.54 | ALL | 303A |
| | PAL20X4A | V05 | 1.54 | ALL | 303A |
| PAL20X10 Exclusive OR | PAL20L10 | V05 | 1.54 | ALL | 303A |
| | PAL20X10 | V05 | 1.54 | ALL | 303A |
| | PAL20X8 | V05 | 1.54 | ALL | 303A |
| | PAL20X4 | V05 | 1.54 | ALL | 303A |
| AmPAL20L10 | AmPAL20L10B/-20/AL | — | — | ALL | 303A-V04 |
| AmPAL20RP10 | AmPAL22P10B/AL/A | — | — | ALL | 303A-V04 |
| | AmPAL20RP10B/AL/A | V10 | — | ALL | 303A-V04 |
| | AmPAL20RP8B/AL/A | V10 | — | ALL | 303A-V04 |
| | AmPAL20RP6B/AL/A | V10 | — | ALL | 303A-V04 |
| | AmPAL20RP4B/AL/A | V10 | — | ALL | 303A-V04 |
| PAL20R8B/B-2*/A/A-2 | PAL20L8B/B-2/A/A-2 | V05 | 1.54 | ALL | 303A |
| | PAL20R8B/B-2/A/A-2 | V05 | 1.54 | ALL | 303A |
| | PAL20R6B/B-2/A/A-2 | V05 | 1.54 | ALL | 303A |
| | PAL20R4B/B-2/A/A-2 | V05 | 1.54 | ALL | 303A |
| PALC20R8Z Zero Standby Power | PALC20L8Z-35/-45 | V10 | 1.54 | ALL | 303A-V04 |
| | PALC20R8Z-35/-45 | V10 | 1.54 | ALL | 303A-V04 |
| | PALC20R6Z-35/-45 | V10 | 1.54 | ALL | 303A-V04 |
| | PALC20R4Z-35/-45 | V10 | 1.54 | ALL | 303A-V04 |
| Decoder | PAL6L16A | V05 | 1.54 | ALL | 303A |
| | PAL8L14A | V05 | 1.54 | ALL | 303A |
| PAL12L10 Combinatorial | PAL12L10 | V05 | 1.54 | ALL | 303A |
| | PAL14L8 | V05 | 1.54 | ALL | 303A |
| | PAL16L6 | V05 | 1.54 | ALL | 303A |
| | PAL18L4 | V05 | 1.54 | ALL | 303A |
| | PAL20L2 | V05 | 1.54 | ALL | 303A |
| | PAL20C1 | V05 | 1.54 | ALL | 303A |
| MegaPAL Device | PAL32R16 | — | 1.54 | 29B | 303A |
| PROSE Device | PMS14R21/A | — | — | ALL | 303A-V04 |
| Programmable Logic Sequencer | PLS105-37 | — | — | ALL | 303A-V04 |
| | PLS167-33 | — | — | ALL | 303A-V04 |
| | PLS168-33 | — | — | ALL | 303A-V04 |
| Fuse Programmable Controller | Am29PL141 | — | — | ALL | 303A-V04 |
| Programmable Event Generator | Am2971 | — | — | ALL | 303A-V04 |
| ECL Registered | PAL10H/10020EV/EG8 | — | — | ALL | 303A-V04 |
| ECL Combinatorial | PAL10H20P8 | — | 1.54 | ALL | 303A-V04 |
| ECL Latched | PAL10H20G8 | — | 1.54 | ALL | 303A-V04 |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Digelec

(201) 493-2420

1.0 System UP803
1.1 P/T Adapter DA53, DA55,DA60, DA62
1.2 Logic Center FAM52
2.0 System 860

### 20 Pin Device Families

| Family | Product | FAM52 | Adapter Rev. | Adapter | System 860 Rev. |
|---|---|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — | -- | — |
| Asynchronous | PAL16RA8 | 5.4 | DA53 | C-1 | A-1.2 |
| PAL16RP8A Programmable Polarity | PAL16P8A | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16RP8A | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16RP6A | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16RP4A | 5.4 | DA53 | A-3 | A-1.2 |
| PAL16R8-10 | PAL16L8-10/H-15 | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R8-10/H-15 | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R6-10/H-15 | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R4-10/H-15 | 5.4 | DA53 | C-1 | A-1.2 |
| PAL16R8D/B | PAL16L8D/B | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R8D/B | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R6D/B | 5.4 | DA53 | C-1 | A-1.2 |
| | PAL16R4D/B | 5.4 | DA53 | C-1 | A-1.2 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L | 5.5 | DA53 | A-3 | — |
| | AmPAL16R8/B/AL/A/Q/L | 5.5 | DA53 | A-3 | — |
| | AmPAL16R6/B/AL/A/Q/L | 5.5 | DA53 | A-3 | — |
| | AmPAL16R4/B/AL/A/Q/L | 5.5 | DA53 | A-3 | — |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4 | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16R8/B-2/B-4/A/A-2/A-4 | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16R6/B-2/B-4/A/A-2/A-4 | 5.4 | DA53 | A-3 | A-1.2 |
| | PAL16R4/B-2/B-4/A/A-2/A-4 | 5.4 | DA53 | A-3 | A-1.2 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25 | — | — | — | — |
| | PALC16R8Q-25 | — | — | — | — |
| | PALC16R6Q-25 | — | — | — | — |
| | PALC16R4Q-25 | — | — | — | — |
| AmPAL16HD8 | AmPAL16H8A/L | 5.5 | DA53 | A-3 | — |
| | AmPAL16HD8A/L | 5.5 | DA53 | A-3 | — |
| | AmPAL16LD8A/L | 5.5 | DA53 | A-3 | — |
| Arithmetic | PAL16X4 | 5.4 | DA53 | A-3 | A-1.2 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | 5.4 | DA55 | B-3 | — |
| PAL10H8 Combinatorial | PAL10H8/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL10L8/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL12H6/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL12L6/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL14H4/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL14L4/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL16H2/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL16L2/-2 | 5.4 | DA53 | A-3 | A-1 |
| | PAL16C1/-2 | 5.4 | DA53 | A-3 | A-1 |

### 24 Pin and MegaPAL Device Families

| Family | Product | FAM52 | Adapter Rev. | Adapter | System 860 Rev. |
|---|---|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — | — | — |
| Varied with XOR | PAL32VX10/A | 6.54 | DA55 | C-1 | A-1.2 |
| Varied Product Terms | AmPAL22V10/-15/A | 5.4 | DA55 | B-3 | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | — | — | — |
| Registered XOR | PAL22RX8A | — | — | — | — |

## Digelec

| Family | Product | FAM52 | Adapter Rev. | Adapter | System 860 Rev. |
|---|---|---|---|---|---|
| **Asynchronous** | PAL20RA10-20 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20RA10 | 5.4 | DA55 | C-1 | A1-2 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L | — | — | — | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | — | — | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | — | — | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | — | — | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | — | — | — |
| **PAL20RS10 Shared Product Terms** | PAL20S10 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20RS10 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20RS8 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20RS4 | 5.4 | DA55 | C-1 | A1-2 |
| **PAL20X10A Exclusive OR** | PAL20L10A | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X10A | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X8A | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X4A | 5.4 | DA55 | C-1 | A1-2 |
| **PAL20X10 Exclusive OR** | PAL20L10 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X10 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X8 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20X4 | 5.4 | DA55 | C-1 | A1-2 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | 5.4 | DA55 | C-1 | A1-2 |
| **AmPAL20RP10** | AmPAL22P10B/AL/A | 5.4 | DA55 | C-1 | A1-2 |
| | AmPAL20RP10B/AL/A | 5.4 | DA55 | C-1 | A1-2 |
| | AmPAL20RP8B/AL/A | 5.4 | DA55 | C-1 | A1-2 |
| | AmPAL20RP6B/AL/A | 5.4 | DA55 | C-1 | A1-2 |
| | AmPAL20RP4B/AL/A | 5.4 | DA55 | C-1 | A1-2 |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20R8B/B-2/A/A-2 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20R6B/B-2/A/A-2 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20R4B/B-2/A/A-2 | 5.4 | DA55 | C-1 | A1-2 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45 | 5.4 | DA55 | C-1 | A1-2 |
| | PALC20R8Z-35/-45 | 5.4 | DA55 | C-1 | A1-2 |
| | PALC20R6Z-35/-45 | 5.4 | DA55 | C-1 | A1-2 |
| | PALC20R4Z-35/-45 | 5.4 | DA55 | C-1 | A1-2 |
| **Decoder** | PAL6L16A | 6.54 | DA62 | — | A1-2 |
| | PAL8L14A | 6.54 | DA62 | — | A1-2 |
| **PAL12L10 Combinatorial** | PAL12L10 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL14L8 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL16L6 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL18L4 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20L2 | 5.4 | DA55 | C-1 | A1-2 |
| | PAL20C1 | 5.4 | DA55 | C-1 | A1-2 |
| **MegaPAL Device** | PAL32R16 | — | — | — | — |
| **PROSE Device** | PMS14R21/A | — | — | — | — |
| **Programmable Logic Sequencer** | PLS105-37 | — | — | — | — |
| | PLS167-33 | — | — | — | — |
| | PLS168-33 | — | — | — | — |
| **Fuse Programmable Controller** | Am29PL141 | — | — | — | — |
| **Programmable Event Generator** | Am2971 | — | — | — | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — | — | — | — |
| **ECL Combinatorial** | PAL10H20P8 | 5.4 | DA60 | A-1 | — |
| **ECL Latched** | PAL10H20G8 | 5.4 | DA60 | A-1 | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Kontron

(415) 965-7020

1.0 System MPP-80S
1.1 Module MOD 21
1.2 Socket Adapter SA-27
1.3 Socket Adapter SA-27-1
2.0 System EPP-80
2.1 Module UPM-B

20 Pin Device Families

| Family | Product | Adapter | UPM-B Rev. |
|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — |
| Asynchronous | PAL16RA8 | — | 1.47 |
| PAL16RP8A Programmable Polarity | PAL16P8A | — | 1.44 |
| | PAL16RP | — | 1.44 |
| | PAL16RP6A | — | 1.44 |
| | PAL16RP4A | — | 1.44 |
| PAL16R8-10 | PAL16L8-10/H-15 | — | 1.44 |
| | PAL16R8-10/H-15 | — | 1.44 |
| | PAL16R6-10/H-15 | — | 1.44 |
| | PAL16R4-10/H-15 | — | 1.44 |
| PAL16R8D/B | PAL16L8D/B | — | 1.44 |
| | PAL16R8D/B | — | 1.44 |
| | PAL16R6D/B | — | 1.44 |
| | PAL16R4D/B | — | 1.44 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L | — | — |
| | AmPAL16R8/B/AL/A/Q/L | — | — |
| | AmPAL16R6/B/AL/A/Q/L | — | — |
| | AmPAL16R4/B/AL/A/Q/L | — | — |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4 | SA-27 | 1.44 |
| | PAL16R8/B-2/B-4/A/A-2/A-4 | SA-27 | 1.44 |
| | PAL16R6/B-2/B-4/A/A-2/A-4 | SA-27 | 1.44 |
| | PAL16R4/B-2/B-4/A/A-2/A-4 | SA-27 | 1.44 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25 | — | — |
| | PALC16R8Q-25 | — | — |
| | PALC16R6Q-25 | — | — |
| | PALC16R4Q-25 | — | — |
| AmPAL16HD8 | AmPAL16H8A/L | — | — |
| | AmPAL16HD8A/L | — | — |
| | AmPAL16LD8A/L | — | — |
| Arithmetic | PAL16X4 | SA-27 | 1.44 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — | — |
| PAL10H8 Combinatorial | PAL10H8/-2 | SA-27 | 1.44 |
| | PAL10L8/-2 | SA-27 | 1.44 |
| | PAL12H6/-2 | SA-27 | 1.44 |
| | PAL12L6/-2 | SA-27 | 1.44 |
| | PAL14H4/-2 | SA-27 | 1.44 |
| | PAL14L4/-2 | SA-27 | 1.44 |
| | PAL16H2/-2 | SA-27 | 1.44 |
| | PAL16L2/-2 | SA-27 | 1.44 |
| | PAL16C1/-2 | SA-27 | 1.44 |

24 Pin and MegaPAL Device Families

| Family | Product | Adapter | UPM-B Rev. |
|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — |
| Varied with XOR | PAL32VX10/A | — | 2.0 |
| Varied Product Terms | AmPAL22V10/-15/A | — | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | 2.0 |
| Registered XOR | PAL22RX8A | — | 2.0 |

## Kontron

| Family | Product | Adapter | UPM-B Rev. |
|---|---|---|---|
| **Asynchronous** | PAL20RA10-20 | — | — |
| | PAL20RA10 | — | 1.44 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | — |
| **PAL20RS10 Shared Product Terms** | PAL20S10 | — | 1.44 |
| | PAL20RS10 | — | 1.44 |
| | PAL20RS8 | — | 1.44 |
| | PAL20RS4 | — | 1.44 |
| **PAL20X10A Exclusive OR** | PAL20L10A | SA-27-1 | 1.44 |
| | PAL20X10A | SA-27-1 | 1.44 |
| | PAL20X8A | SA-27-1 | 1.44 |
| | PAL20X4A | SA-27-1 | 1.44 |
| **PAL20X10 Exclusive OR** | PAL20L10 | SA-27-1 | 1.44 |
| | PAL20X10 | SA-27-1 | 1.44 |
| | PAL20X8 | SA-27-1 | 1.44 |
| | PAL20X4 | SA-27-1 | 1.44 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A | — | — |
| | AmPAL20RP10B/AL/A | — | — |
| | AmPAL20RP8B/AL/A | — | — |
| | AmPAL20RP6B/AL/A | — | — |
| | AmPAL20RP4B/AL/A | — | — |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2 | SA-27-1 | 1.44 |
| | PAL20R8B/B-2/A/A-2 | SA-27-1 | 1.44 |
| | PAL20R6B/B-2/A/A-2 | SA-27-1 | 1.44 |
| | PAL20R4B/B-2/A/A-2 | SA-27-1 | 1.44 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45 | — | 2.00 |
| | PALC20R8Z-35/-45 | — | 2.00 |
| | PALC20R6Z-35/-45 | — | 2.00 |
| | PALC20R4Z-35/-45 | — | 2.00 |
| **Decoder** | PAL6L16A | — | 2.00 |
| | PAL8L14A | — | 2.00 |
| **PAL12L10 Combinatorial** | PAL12L10 | SA-27-1 | 1.44 |
| | PAL14L8 | SA-27-1 | 1.44 |
| | PAL16L6 | SA-27-1 | 1.44 |
| | PAL18L4 | SA-27-1 | 1.44 |
| | PAL20L2 | SA-27-1 | 1.44 |
| | PAL20C1 | SA-27-1 | 1.44 |
| **MegaPAL Device** | PAL32R16 | — | — |
| **PROSE Device** | PMS14R21/A | — | — |
| **Programmable Logic Sequencer** | PLS105-37 | — | — |
| | PLS167-33 | — | — |
| | PLS168-33 | — | — |
| **Fuse Programmable Controller** | Am29PL141 | — | — |
| **Programmable Event Generator** | Am2971 | — | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — | — |
| **ECL Combinatorial ECL Latched** | PAL10H20P8 | — | 1.47 |
| | PAL10H20G8 | — | 1.47 |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Logical Devices, Inc.

1.0 System ALLPRO

1321 E. Northwest
65th Place
Fort Lauderdale, FL 33309
(800) 331-7766

### 20 Pin Device Families

| Family | Product | Software |
|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — |
| Asynchronous | PAL16RA8 | 1.44CR2 |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | 1.44CR2<br>1.44CR2<br>1.44CR2 |
| Arithmetic | PAL16X4 | 1.44CR2 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |

### 24 Pin and MegaPAL Device Families

| Family | Product | Software |
|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — |
| Varied with XOR | PAL32VX10/A | 1.44CR2 |
| Varied Product Terms | AmPAL22V10/-15/A | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — |
| Registered XOR | PAL22RX8A | 1.44CR2 |

## Logical Devices, Inc.

| Family | Product | Software |
|---|---|---|
| **Asynchronous** | PAL20RA10-20<br>PAL20RA10 | 1.44CR2<br>1.44CR2 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L<br>AmPAL20XRP10-20/-30L/-30/-40L<br>AmPAL20XRP8-20/-30L/-30/-40L<br>AmPAL20XRP6-20/-30L/-30/-40L<br>AmPAL20XRP4-20/-30L/-30/-40L | —<br>—<br>—<br>—<br>— |
| **PAL20RS10 Shared Product Terms** | PAL20S10<br>PAL20RS10<br>PAL20RS8<br>PAL20RS4 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **PAL20X10A Exclusive OR** | PAL20L10A<br>PAL20X10A<br>PAL20X8A<br>PAL20X4A | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **PAL20X10 Exclusive OR** | PAL20L10<br>PAL20X10<br>PAL20X8<br>PAL20X4 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A<br>AmPAL20RP10B/AL/A<br>AmPAL20RP8B/AL/A<br>AmPAL20RP6B/AL/A<br>AmPAL20RP4B/AL/A | —<br>—<br>—<br>— |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2<br>PAL20R8B/B-2/A/A-2<br>PAL20R6B/B-2/A/A-2<br>PAL20R4B/B-2/A/A-2 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45<br>PALC20R8Z-35/-45<br>PALC20R6Z-35/-45<br>PALC20R4Z-35/-45 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **Decoder** | PAL6L16A<br>PAL8L14A | 1.44CR2<br>1.44CR2 |
| **PAL12L10 Combinatorial** | PAL12L10<br>PAL14L8<br>PAL16L6<br>PAL18L4<br>PAL20L2<br>PAL20C1 | 1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2<br>1.44CR2 |
| **MegaPAL Device** | PAL32R16 | — |
| **PROSE Device** | PMS14R21/A | — |
| **Programmable Logic Sequencer** | PLS105-37<br>PLS167-33<br>PLS168-33 | —<br>—<br>— |
| **Fuse Programmable Controller** | Am29PL141 | — |
| **Programmable Event Generator** | Am2971 | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — |
| **ECL Combinatorial**<br>**ECL Latched** | PAL10H20P8<br>PAL10H20G8 | —<br>— |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products. Later software and hardware revisions can be assumed to support these products.

## Microprоss

1.0 ROM 5000

204-79040
France

### 20 Pin Device Families

| Family | Product | Software Rev. |
|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — |
| Asynchronous | PAL16RA8 | 4.6 |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 3.5<br>3.5<br>3.5<br>3.5 |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | 3.5<br>3.5<br>3.5<br>3.5 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | 3.5<br>3.5<br>3.5<br>3.5 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | —<br>—<br>—<br>— |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 3.5<br>3.5<br>3.5<br>3.5 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | —<br>—<br>— |
| Arithmetic | PAL16X4 | 3.5 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 3.5<br>3.5<br>3.5<br>3.5<br>3.5<br>3.5<br>3.5<br>3.5<br>3.5 |

### 24 Pin and MegaPAL Device Families

| Family | Product | Software Rev. |
|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — |
| Varied with XOR | PAL32VX10/A | 4.51 |
| Varied Product Terms | AmPAL22V10/-15/A | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | 4.6 |
| Registered XOR | PAL22RX8A | — |

## Microprocess

| Family | Product | Software Rev. |
|---|---|---|
| **Asynchronous** | PAL20RA10-20<br>PAL20RA10 | 3.5<br>3.5 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L<br>AmPAL20XRP10-20/-30L/-30/-40L<br>AmPAL20XRP8-20/-30L/-30/-40L<br>AmPAL20XRP6-20/-30L/-30/-40L<br>AmPAL20XRP4-20/-30L/-30/-40L | —<br>—<br>—<br>—<br>— |
| **PAL20RS10 Shared Product Terms** | PAL20S10<br>PAL20RS10<br>PAL20RS8<br>PAL20RS4 | 3.5<br>3.5<br>3.5<br>3.5 |
| **PAL20X10A Exclusive OR** | PAL20L10A<br>PAL20X10A<br>PAL20X8A<br>PAL20X4A | 3.5<br>3.5<br>3.5<br>3.5 |
| **PAL20X10 Exclusive OR** | PAL20L10<br>PAL20X10<br>PAL20X8<br>PAL20X4 | 3.5<br>3.5<br>3.5<br>3.5 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A<br>AmPAL20RP10B/AL/A<br>AmPAL20RP8B/AL/A<br>AmPAL20RP6B/AL/A<br>AmPAL20RP4B/AL/A | —<br>—<br>—<br>—<br>— |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2<br>PAL20R8B/B-2/A/A-2<br>PAL20R6B/B-2/A/A-2<br>PAL20R4B/B-2/A/A-2 | 3.5<br>3.5<br>3.5<br>3.5 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45<br>PALC20R8Z-35/-45<br>PALC20R6Z-35/-45<br>PALC20R4Z-35/-45 | —<br>—<br>—<br>— |
| **Decoder** | PAL6L16A<br>PAL8L14A | 4.6<br>4.6 |
| **PAL12L10 Combinatorial** | PAL12L10<br>PAL14L8<br>PAL16L6<br>PAL18L4<br>PAL20L2<br>PAL20C1 | 3.5<br>3.5<br>3.5<br>3.5<br>3.5<br>3.5 |
| **MegaPAL Device** | PAL32R16 | — |
| **PROSE Device** | PMS14R21/A | — |
| **Programmable Logic Sequencer** | PLS105-37<br>PLS167-33<br>PLS168-33 | —<br>—<br>— |
| **Fuse Programmable Controller** | Am29PL141 | — |
| **Programmable Event Generator** | Am2971 | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — |
| **ECL Combinatorial ECL Latched** | PAL10H20P8<br>PAL10H20G8 | 4.6<br>4.6 |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Stag Microsystems

(408) 988-1118

1.0 ZL 30
2.0 PPZ
2.1 Module ZM2200

20 Pin Device Families

| Family | Product | Code | ZL 30 Rev. | ZM2200 Rev. |
|---|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — | — |
| Asynchronous | PAL16RA8 | 20-19 | 30-37 | 15 |
| PAL16RP8A Programmable Polarity | PAL16P8A | 20-38 | 30-35 | 15 |
| | PAL16RP8A | 20-11 | 30-35 | 15 |
| | PAL16RP6A | 20-12 | 30-35 | 15 |
| | PAL16RP4A | 20-13 | 30-35 | 15 |
| PAL16R8-10 | PAL16L8-10/H-15 | 22-29 | 30-39 | 12 |
| | PAL16R8-10/H-15 | 22-30 | 30-39 | 12 |
| | PAL16R6-10/H-15 | 22-31 | 30-39 | 12 |
| | PAL16R4-10/H-15 | 22-32 | 30-39 | 12 |
| PAL16R8D/B | PAL16L8D/B | 22-29 | 30-39 | 12 |
| | PAL16R8D/B | 22-30 | 30-39 | 12 |
| | PAL16R6D/B | 22-31 | 30-39 | 12 |
| | PAL16R4D/B | 22-32 | 30-39 | 12 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L | 90-29 | 30-35 | 14 |
| | AmPAL16R8/B/AL/A/Q/L | 90-30 | 30-35 | 14 |
| | AmPAL16R6/B/AL/A/Q/L | 90-31 | 30-35 | 14 |
| | AmPAL16R4/B/AL/A/Q/L | 90-32 | 30-35 | 14 |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4 | 20-29 | 30-35 | 14 |
| | PAL16R8/B-2/B-4/A/A-2/A-4 | 20-30 | 30-35 | 14 |
| | PAL16R6/B-2/B-4/A/A-2/A-4 | 20-31 | 30-35 | 14 |
| | PAL16R4/B-2/B-4/A/A-2/A-4 | 20-32 | 30-35 | 14 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25 | — | — | — |
| | PALC16R8Q-25 | — | — | — |
| | PALC16R6Q-25 | — | — | — |
| | PALC16R4Q-25 | — | — | — |
| AmPAL16HD8 | AmPAL16H8A/L | 90-35 | 30-35 | 14 |
| | AmPAL16HD8A/L | 90-37 | 30-35 | 14 |
| | AmPAL16LD8A/L | 90-36 | 30-35 | 14 |
| Arithmetic | PAL16X4 | 20-33 | 30-35 | 14 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | 90-10 | 30-38 | 19 |
| PAL10H8 Combinatorial | PAL10H8/-2 | 20-20 | 30-35 | 14 |
| | PAL10L8/-2 | 20-25 | 30-35 | 14 |
| | PAL12H6/-2 | 20-21 | 30-35 | 14 |
| | PAL12L6/-2 | 20-26 | 30-35 | 14 |
| | PAL14H4/-2 | 20-22 | 30-35 | 14 |
| | PAL14L4/-2 | 20-27 | 30-35 | 14 |
| | PAL16H2/-2 | 20-23 | 30-35 | 14 |
| | PAL16L2/-2 | 20-28 | 30-35 | 14 |
| | PAL16C1/-2 | 20-24 | 30-35 | 14 |

24 Pin and MegaPAL Device Families

| Family | Product | Code | ZL 30 Rev. | ZM2200 Rev. |
|---|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — | — |
| Varied with XOR | PAL32VX10/A | 21-66 | — | 23 |
| Varied Product Terms | AmPAL22V10/-15/A | 91-70 | 30-38 | 10 |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | — | — |
| Registered XOR | PAL22RX8A | 21-98 | — | 23 |

## Stag Microsystems

| Family | Product | Code | ZL 30 Rev. | ZM2200 Rev. |
|---|---|---|---|---|
| Asynchronous | PAL20RA10-20 | — | — | — |
| | PAL20RA10 | 21-77 | 30-37 | 15 |
| AmPAL20XRP10 | AmPAL22XP10-20/-30L/-30/-40L | — | — | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | — | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | — | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | — | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | — | — |
| PAL20RS10 Shared Product Terms | PAL20S10 | 21-81 | 30-39 | 15 |
| | PAL20RS10 | 21-80 | 30-39 | 15 |
| | PAL20RS8 | 21-79 | 30-39 | 15 |
| | PAL20RS4 | 21-78 | 30-39 | 15 |
| PAL20X10A Exclusive OR | PAL20L10A | 21-60 | 30-35 | 12 |
| | PAL20X10A | 21-61 | 30-35 | 12 |
| | PAL20X8A | 21-62 | 30-35 | 12 |
| | PAL20X4A | 21-63 | 30-35 | 12 |
| PAL20X10 Exclusive OR | PAL20L10 | 21-60 | 30-35 | 12 |
| | PAL20X10 | 21-61 | 30-35 | 12 |
| | PAL20X8 | 21-62 | 30-35 | 12 |
| | PAL20X4 | 21-63 | 30-35 | 12 |
| AmPAL20L10 | AmPAL20L10B/-20/AL | — | — | — |
| AmPAL20RP10 | AmPAL22P10B/AL/A | — | — | — |
| | AmPAL20RP10B/AL/A | — | — | — |
| | AmPAL20RP8B/AL/A | — | — | — |
| | AmPAL20RP6B/AL/A | — | — | — |
| | AmPAL20RP4B/AL/A | — | — | — |
| PAL20R8B/B-2/A/A-2 | PAL20L8B/B-2/A/A-2 | 21-56 | 30-35 | 12 |
| | PAL20R8B/B-2/A/A-2 | 21-57 | 30-35 | 12 |
| | PAL20R6B/B-2/A/A-2 | 21-58 | 30-35 | 12 |
| | PAL20R4B/B-2/A/A-2 | 21-59 | 30-35 | 12 |
| PALC20R8Z Zero Standby Power | PALC20L8Z-35/-45 | 24-56 | — | 23 |
| | PALC20R8Z-35/-45 | 24-57 | — | 23 |
| | PALC20R6Z-35/-45 | 24-58 | — | 23 |
| | PALC20R4Z-35/-45 | 24-59 | — | 23 |
| Decoder | PAL6L16A | — | — | — |
| | PAL8L14A | — | — | — |
| PAL12L10 Combinatorial | PAL12L10 | 21-50 | 30-35 | 14 |
| | PAL14L8 | 21-51 | 30-35 | 14 |
| | PAL16L6 | 21-52 | 30-35 | 12 |
| | PAL18L4 | 21-53 | 30-35 | 12 |
| | PAL20L2 | 21-54 | 30-35 | 12 |
| | PAL20C1 | 21-55 | 30-35 | 12 |
| MegaPAL Device | PAL32R16 | — | — | — |
| PROSE Device | PMS14R21/A | — | — | — |
| Programmable Logic Sequencer | PLS105-37 | — | — | — |
| | PLS167-33 | — | — | — |
| | PLS168-33 | — | — | — |
| Fuse Programmable Controller | Am29PL141 | — | — | — |
| Programmable Event Generator | Am2971 | — | — | — |
| ECL Registered | PAL10H/10020EV/EG8 | — | — | — |
| ECL Combinatorial | PAL10H20P8 | — | — | — |
| ECL Latched | PAL10H20G8 | — | — | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Storey Systems

1.0 P240

(214) 270-4135

### 20 Pin Device Families

| Family | Product | Rev. |
|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — |
| Asynchronous | PAL16RA8 | 4.04 |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 4.0<br>4.0<br>4.0 |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | 4.0<br>4.0<br>4.0<br>4.0 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | 4.0<br>4.0<br>4.0<br>4.0 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | —<br>—<br>—<br>— |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 2.0<br>2.0<br>2.0<br>2.0 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | —<br>—<br>— |
| Arithmetic | PAL16X4 | 2.0 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 2.0<br>2.0<br>2.0<br>2.0<br>2.0<br>2.0<br>2.0<br>2.0<br>2.0 |

### 24 Pin and MegaPAL Device Families

| Family | Product | Rev. |
|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — |
| Varied with XOR | PAL32VX10/A | — |
| Varied Product Terms | AmPAL22V10/-15/A | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — |
| Registered XOR | PAL22RX8A | — |

## Storey Systems

| Family | Product | Rev. |
|---|---|---|
| **Asynchronous** | PAL20RA10-20 | — |
| | PAL20RA10 | 4.04 |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — |
| **PAL20RS10 Shared Product Terms** | PAL20S10 | — |
| | PAL20RS10 | — |
| | PAL20RS8 | — |
| | PAL20RS4 | — |
| **PAL20X10A Exclusive OR** | PAL20L10A | 2.0 |
| | PAL20X10A | 2.0 |
| | PAL20X8A | 2.0 |
| | PAL20X4A | 2.0 |
| **PAL20X10 Exclusive OR** | PAL20L10 | 2.0 |
| | PAL20X10 | 2.0 |
| | PAL20X8 | 2.0 |
| | PAL20X4 | 2.0 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A | — |
| | AmPAL20RP10B/AL/A | — |
| | AmPAL20RP8B/AL/A | — |
| | AmPAL20RP6B/AL/A | — |
| | AmPAL20RP4B/AL/A | — |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2 | 2.0 |
| | PAL20R8B/B-2/A/A-2 | 2.0 |
| | PAL20R6B/B-2/A/A-2 | 2.0 |
| | PAL20R4B/B-2/A/A-2 | 2.0 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45 | — |
| | PALC20R8Z-35/-45 | — |
| | PALC20R6Z-35/-45 | — |
| | PALC20R4Z-35/-45 | — |
| **Decoder** | PAL6L16A | — |
| | PAL8L14A | — |
| **PAL12L10 Combinatorial** | PAL12L10 | 2.0 |
| | PAL14L8 | 2.0 |
| | PAL16L6 | 2.0 |
| | PAL18L4 | 2.0 |
| | PAL20L2 | 2.0 |
| | PAL20C1 | 2.0 |
| **MegaPAL Device** | PAL32R16 | — |
| **PROSE Device** | PMS14R21/A | — |
| **Programmable Logic Sequencer** | PLS105-37 | — |
| | PLS167-33 | — |
| | PLS168-33 | — |
| **Fuse Programmable Controller** | Am29PL141 | — |
| **Programmable Event Generator** | Am2971 | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — |
| **ECL Combinatorial ECL Latched** | PAL10H20P8 | — |
| | PAL10H20G8 | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Structured Design

1.0 SD 20/24
2.0 SD 1000

(408) 988-0725

### 20 Pin Device Families

| Family | Product | SD 20/24 Rev. | SD 1000 Rev. |
|---|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — | — |
| Asynchronous | PAL16RA8 | — | — |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | —<br>—<br>—<br>— | —<br>—<br>—<br>— |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | —<br>—<br>—<br>— | 1.05<br>1.05<br>1.05<br>1.05 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | —<br>—<br>—<br>— | 1.05<br>1.05<br>1.05<br>1.05 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | —<br>—<br>—<br>— | 1.05<br>1.05<br>1.05<br>1.05 |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 1.6<br>1.6<br>1.6<br>1.6 | 1.05<br>1.05<br>1.05<br>1.05 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | —<br>—<br>— | 1.05<br>1.05<br>1.05 |
| Arithmetic | PAL16X4 | 1.6 | 1.05 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — | 1.05 |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 1.6<br>1.6<br>1.6<br>1.6<br>1.6<br>1.6<br>1.6<br>1.6<br>1.6 | 1.05<br>1.05<br>1.05<br>1.05<br>1.05<br>1.05<br>1.05<br>1.05<br>1.05 |

### 24 Pin and MegaPal Device Families

| Family | Product | SD 20/24 Rev. | SD 1000 Rev. |
|---|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — | — |
| Varied with XOR | PAL32VX10/A | — | — |
| Varied Product Terms | AmPAL22V10/-15/A | — | 1.05 |
| Varied Terms (CMOS) | PALC22V10H-25/35 | — | — |
| Registered XOR | PAL22RX8A | — | — |

## Structured Design

| Family | Product | SD 20/24 Rev. | SD 1000 Rev. |
|---|---|---|---|
| **Asynchronous** | PAL20RA10-20 | — | — |
| | PAL20RA10 | — | — |
| **AmPAL20XRP10** | AmPAL22XP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — | — |
| **PAL20RS10 Shared Product Terms** | PAL20S10 | — | — |
| | PAL20RS10 | — | — |
| | PAL20RS8 | — | — |
| | PAL20RS4 | — | — |
| **PAL20X10A Exclusive OR** | PAL20L10A | 1.6 | 1.05 |
| | PAL20X10A | 1.6 | 1.05 |
| | PAL20X8A | 1.6 | 1.05 |
| | PAL20X4A | 1.6 | 1.05 |
| **PAL20X10 Exclusive OR** | PAL20L10 | 1.6 | 1.05 |
| | PAL20X10 | 1.6 | 1.05 |
| | PAL20X8 | 1.6 | 1.05 |
| | PAL20X4 | 1.6 | 1.05 |
| **AmPAL20L10** | AmPAL20L10B/-20/AL | — | — |
| **AmPAL20RP10** | AmPAL22P10B/AL/A | — | — |
| | AmPAL20RP10B/AL/A | — | — |
| | AmPAL20RP8B/AL/A | — | — |
| | AmPAL20RP6B/AL/A | — | — |
| | AmPAL20RP4B/AL/A | — | — |
| **PAL20R8B/B-2/A/A-2** | PAL20L8B/B-2/A/A-2 | 1.6 | 1.05 |
| | PAL20R8B/B-2/A/A-2 | 1.6 | 1.05 |
| | PAL20R6B/B-2/A/A-2 | 1.6 | 1.05 |
| | PAL20R4B/B-2/A/A-2 | 1.6 | 1.05 |
| **PALC20R8Z Zero Standby Power** | PALC20L8Z-35/-45 | — | — |
| | PALC20R8Z-35/-45 | — | — |
| | PALC20R6Z-35/-45 | — | — |
| | PALC20R4Z-35/-45 | — | — |
| **Decoder** | PAL6L16A | — | — |
| | PAL8L14A | — | — |
| **PAL12L10 Combinatorial** | PAL12L10 | 1.6 | 1.05 |
| | PAL14L8 | 1.6 | 1.05 |
| | PAL16L6 | 1.6 | 1.05 |
| | PAL18L4 | 1.6 | 1.05 |
| | PAL20L2 | 1.6 | 1.05 |
| | PAL20C1 | 1.6 | 1.05 |
| **MegaPAL Device** | PAL32R16 | — | — |
| **PROSE Device** | PMS14R21/A | — | — |
| **Programmable Logic Sequencer** | PLS105-37 | — | — |
| | PLS167-33 | — | — |
| | PLS168-33 | — | — |
| **Fuse Programmable Controller** | Am29PL141 | — | — |
| **Programmable Event Generator** | Am2971 | — | — |
| **ECL Registered** | PAL10H/10020EV/EG8 | — | — |
| **ECL Combinatorial ECL Latched** | PAL10H20P8 | — | — |
| | PAL10H20G8 | — | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

## Varix

(214) 437-0777

1.0 Omni Programmer
1.1 Adapter HVX5A-B01
1.2 Adapter A01

### 20 Pin Device Families

| Family | Product | Software Rev. |
|---|---|---|
| Sequencer | AmPAL23S8-20/-25 | — |
| Asynchronous | PAL16RA8 | — |
| PAL16RP8A Programmable Polarity | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 3.18<br>3.18<br>3.18<br>3.18 |
| PAL16R8-10 | PAL16L8-10/H-15<br>PAL16R8-10/H-15<br>PAL16R6-10/H-15<br>PAL16R4-10/H-15 | 3.18<br>3.18<br>3.18<br>3.18 |
| PAL16R8D/B | PAL16L8D/B<br>PAL16R8D/B<br>PAL16R6D/B<br>PAL16R4D/B | 3.18<br>3.18<br>3.18<br>3.18 |
| AmPAL16R8 | AmPAL16L8/B/AL/A/Q/L<br>AmPAL16R8/B/AL/A/Q/L<br>AmPAL16R6/B/AL/A/Q/L<br>AmPAL16R4/B/AL/A/Q/L | —<br>—<br>—<br>— |
| PAL16R8/B-2/B-4/ A/A-2/A-4 | PAL16L8/B-2/B-4/A/A-2/A-4<br>PAL16R8/B-2/B-4/A/A-2/A-4<br>PAL16R6/B-2/B-4/A/A-2/A-4<br>PAL16R4/B-2/B-4/A/A-2/A-4 | 3.18<br>3.18<br>3.18<br>3.18 |
| PALC16R8Q-25 (CMOS) | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | —<br>—<br>—<br>— |
| AmPAL16HD8 | AmPAL16H8A/L<br>AmPAL16HD8A/L<br>AmPAL16LD8A/L | —<br>—<br>— |
| Arithmetic | PAL16X4 | 3.18 |
| Combinatorial | AmPAL18P8B/AL/A/Q/L | — |
| PAL10H8 Combinatorial | PAL10H8/-2<br>PAL10L8/-2<br>PAL12H6/-2<br>PAL12L6/-2<br>PAL14H4/-2<br>PAL14L4/-2<br>PAL16H2/-2<br>PAL16L2/-2<br>PAL16C1/-2 | 3.18<br>3.18<br>3.18<br>3.18<br>3.18<br>3.18<br>3.18<br>3.18<br>3.18 |

### 24 Pin and MegaPAL Device Families

| Family | Product | Software Rev. |
|---|---|---|
| Macrocell (Async) | AmPALC29MA16-35/-45 | — |
| Macrocell (Sync) | AmPALC29M16-35/-45 | — |
| Varied with XOR | PAL32VX10/A | 5.00A |
| Varied Product Terms | AmPAL22V10/-15/A | — |
| Varied Terms (CMOS) | PALC22V10H-25/35 | 5.00A |
| Registered XOR | PAL22RX8A | 5.00A |

## Varix

| Family | Product | Software Rev. |
|---|---|---|
| Asynchronous | PAL20RA10-20 | — |
| | PAL20RA10 | 3.18 |
| AmPAL20XRP10 | AmPAL22XP10-20/-30L/-30/-40L | — |
| | AmPAL20XRP10-20/-30L/-30/-40L | — |
| | AmPAL20XRP8-20/-30L/-30/-40L | — |
| | AmPAL20XRP6-20/-30L/-30/-40L | — |
| | AmPAL20XRP4-20/-30L/-30/-40L | — |
| PAL20RS10 Shared Product Terms | PAL20S10 | — |
| | PAL20RS10 | — |
| | PAL20RS8 | — |
| | PAL20RS4 | — |
| PAL20X10A Exclusive OR | PAL20L10A | 3.18 |
| | PAL20X10A | 3.18 |
| | PAL20X8A | 3.18 |
| | PAL20X4A | 3.18 |
| PAL20X10 Exclusive OR | PAL20L10 | 3.18 |
| | PAL20X10 | 3.18 |
| | PAL20X8 | 3.18 |
| | PAL20X4 | 3.18 |
| AmPAL20L10 | AmPAL20L10B/-20/AL | — |
| AmPAL20RP10 | AmPAL22P10B/AL/A | — |
| | AmPAL20RP10B/AL/A | — |
| | AmPAL20RP8B/AL/A | — |
| | AmPAL20RP6B/AL/A | — |
| | AmPAL20RP4B/AL/A | — |
| PAL20R8B/B-2/A/A-2 | PAL20L8B/B-2/A/A-2 | 3.18 |
| | PAL20R8B/B-2/A/A-2 | 3.18 |
| | PAL20R6B/B-2/A/A-2 | 3.18 |
| | PAL20R4B/B-2/A/A-2 | 3.18 |
| PALC20R8Z Zero Standby Power | PALC20L8Z-35/-45 | 5.0 |
| | PALC20R8Z-35/-45 | 5.0 |
| | PALC20R6Z-35/-45 | 5.0 |
| | PALC20R4Z-35/-45 | 5.0 |
| Decoder | PAL6L16A | 3.18 |
| | PAL8L14A | 3.18 |
| PAL12L10 Combinatorial | PAL12L10 | 3.18 |
| | PAL14L8 | 3.18 |
| | PAL16L6 | 3.18 |
| | PAL18L4 | 3.18 |
| | PAL20L2 | 3.18 |
| | PAL20C1 | 3.18 |
| MegaPAL Device | PAL32R16 | — |
| PROSE Device | PMS14R21/A | 5.00A |
| Programmable Logic Sequencer | PLS105-37 | — |
| | PLS167-33 | — |
| | PLS168-33 | — |
| Fuse Programmable Controller | Am29PL141 | — |
| Programmable Event Generator | Am2971 | — |
| ECL Registered | PAL10H/10020EV/EG8 | — |
| ECL Combinatorial ECL Latched | PAL10H20P8 | — |
| | PAL10H20G8 | — |

Notes: "—" = Contact programmer manufacturer.
The software and hardware revisions listed are the earliest revisions that support these products.
Later software and hardware revisions can be assumed to support these products.

# ProPAL™, HAL® and ZHAL™ Devices Program

ProPAL, HAL and ZHAL devices are programmable logic devices that are programmed, marked and functionally tested by Monolithic Memories. Our functional testing offers the user board-ready product at quality levels as stringent as 50 Parts Per Million (PPM), providing significant benefits in both quality and manufacturing cost savings. The ProPAL, HAL and ZHAL device program provides system manufacturers a risk-free migration path from system prototype to full production with extremely high-quality, board-ready devices.

## ProPAL Devices

ProPAL (Programmed PAL) devices are simply PAL devices that Monolithic Memories programs and tests for you. You receive a fully functional device without having to do any programming and testing, and still have the flexibility to handle design changes easily.

## HAL Devices

HAL (Hard Array Logic) devices are to PAL devices as ROMs are to PROMs. Instead of fuses in the logic array, your pattern is implemented using metal links that are masked in during wafer fabrication.

## ZHAL Devices

ZHAL devices are Zero-Standby-Power CMOS HAL devices. These devices can implement any pattern from our standard and combinatorial 20-pin and 24-pin PAL device families with the greatly reduced power consumption only CMOS can offer.

All ZHAL devices are fully HC/HCT compatible, making them easy to use in TTL and CMOS environments.

## Should You Use a ProPAL, HAL or ZHAL Device?

PAL devices offer the flexibility and convenience needed for prototyping your innovative designs. They provide a means for designing an efficient system by integrating functions and saving board space. For design, prototyping and low-volume production, it makes sense to program and test your own PAL devices. You always have the option of making last-minute design tweaks as you fine-tune your system design.

Once your production volumes begin to ramp up to higher volumes our ProPAL, HAL and ZHAL device offerings provide a cost-effective solution.

At modest initial volumes, ProPAL devices provide the best solution by eliminating programming and testing needs while retaining enough flexibility to accommodate design changes. We offer three different testing options for ProPAL devices, which are described below. A detailed technical description of what these testing options involve follows in the functional testing section.

The AutoVec™ option provides a cost-effective solution for low-volume business (as few as 250 devices/pattern) at a typical quality level of 500-700 PPM.

When your volumes reach a moderate volume of a few thousand devices per year for each pattern, straight-functional or AC-functional tested ProPAL devices provide the right solution. Straight-functional testing provides typical quality levels of 200-400 PPM, and AC-functional testing provides 50 PPM level quality. Of course the AutoVec option is available for these larger volumes, but larger volume business is usually handled with the straight-functional or AC-functional testing options.

When your design has stabilized and your production volume has ramped up to several thousand devices per year, HAL or ZHAL devices are the most cost-effective way to purchase your programmable logic. All HAL and ZHAL devices are fully AC-functional tested and have final quality levels of better than 50 PPM.

The quality levels provided by the various ProPAL, HAL and ZHAL device options are summarized in Table 1.

| TESTING OPTION / DEVICE TYPE | AUTOVEC™ | STRAIGHT FUNCTIONAL | AC-FUNCTIONAL |
|---|---|---|---|
| ProPAL Devices | 500–700 PPM | 200–400 PPM | 50 PPM |
| HAL Devices | N/A | N/A | 50 PPM |
| ZHAL Devices | N/A | N/A | 50 PPM |

**Table 1. Quality Levels for the Various Testing Alternatives in the ProPAL, HAL and ZHAL Devices Program.**

## Quality and Cost Savings

The quality and cost savings benefits the ProPAL, HAL and ZHAL device program offers are substantial. The investment the PAL device user makes in ProPAL, HAL or ZHAL devices yields a significant return in product quality and manufacturing savings.

The quality is a result of our many years and millions of units of experience in the design, manufacture, programming and testing of PAL devices. This experience lets us provide PAL device users finished quality levels as stringent as 50 PPM.

The reduced manufacturing costs are derived from the high quality provided by the ProPAL, HAL and ZHAL device (quality that results in increased manufacturing yield) and reduced component processing and handling costs.

The manufacturing yield (the number of working systems produced as a percentage of total number of systems produced) is a function of the quality of the components in a system. Increasing the quality of the components naturally increases the manufacturing yield. Table 2 is a tabulation of values demonstrating the relationship between component quality and manufacturing yield.

Additional manufacturing cost savings come in reduced processing and handling costs. Purchasing pre-programmed and functionally tested devices direct from the factory eliminates the need for the user to perform any programming or functional testing. This eliminates the need for the user to carry all the associated overhead:

- Programming (programmer equipment cost, floor space, maintenance and calibration, operator costs)
- Labeling/stripping/marking (equipment cost, floor space, maintenance, operator costs)
- Vector generation (computer/software costs, engineering costs)
- Testing (equipment costs, operator costs)
- Elimination of sockets (which allows for auto-insertion)

Also, when ProPAL, HAL or ZHAL devices are utilized, handling-related rejects are virtually eliminated. We have found handling errors to be one of the largest sources of rejects. Mixed device types, mixed bit patterns, mixed reject and good devices, bent leads and ESD damaged devices can all result in board-level failures, failures that can be avoided with ProPAL, HAL or ZHAL devices. ProPAL, HAL and ZHAL devices come programmed, marked and fully tested, ready to go directly from the shipping box to the production floor.

## A Cost Savings Example

A systems manufacturer produces 2000 systems per month, each having 10 PAL devices (total 20,000 PAL devices/month). This manufacturer's cost of diagnosing and reworking a non-functional system (at system test) is $150. What cost benefits does the ProPAL, HAL and ZHAL devices program offer this manufacturer?

### Case I

No functional testing is performed—5000 PPM board-level quality.

Manufacturing yield (from Table 2) = 95.1%

4.9% or 98 systems/month require rework.

At $150/system, total rework costs/month are:

98 x $150 = $14,700.

At 20,000 devices/month, rework costs are:

$14,700/20,000 = $0.73/device.

### Case II

AutoVec testing is performed—500 PPM board-level quality.

Manufacturing yield (from Table 1) = 99.5%.

0.5% or 10 systems/month require rework.

At $150/system, total rework costs/month are:

10 x $150 = $1500.

At 20,000 devices/month, rework costs are:

$1500/20,000 = $0.08/device.

| DEVICES/SYSTEM / PPM | 5 | 10 | 25 | 50 | 100 |
|---|---|---|---|---|---|
| 50 | 99.975% | 99.95% | 99.87% | 99.75% | 99.5% |
| 500 | 99.75% | 99.5% | 98.8% | 97.5% | 95% |
| 5000 | 97.5% | 95.1% | 88% | 78% | 61% |
| 10,000 | 95.1% | 91.5% | 78% | 61% | 37% |

Table 2. Manufacturing Yield for Various Component Quality Levels

### Case III

AC-functional testing is performed—50 PPM board-level quality.

Manufacturing yield (from Table 2) = 99.975%.

0.025% or < 1 system/month requires rework.

At $150/system, total rework costs/month are:

< $150.

At 20,000 devices/month, rework costs are:

< $150/20,000 = $0.01/device.

**Cost Savings** (utilizing):

Autovec testing: $0.73 – $0.08 = $0.65/device.

AC-functional testing: $0.73 – $0.01 = $0.72/device.

This example does not take into account a few things:

- The cost of the manufacturer doing the programming.
- Additional manufacturing cost savings (e.g. elimination of sockets allowing auto-insertion).
- The cost adder for ProPAL, HAL and ZHAL device services.

The cost of doing programming varies from manufacturer to manufacturer. Generally, unless a manufacturer does high volume programming, this cost can be substantial.

Additionally, other manufacturing cost benefits are also realized. If PAL devices are being socketed, the high quality levels of ProPAL, HAL and ZHAL devices allow for elimination of those sockets and the utilization of auto-insertion in the manufacturing flow. Board-ready devices also lend themselves more readily to just-in-time or ship-to-stock purchasing and manufacturing programs.

The cost of ProPAL, HAL and ZHAL device services depends on a number of factors (device type, volume, device base price). It is typically much less than it costs a manufacturer to program and test devices, not to mention the savings realized through the increased quality.

## The Importance of Functional Testing

Programming is final manufacturing, and the quality of a programmed device must be verified by thorough testing. After programming, a device is "array verified." This verification checks the fuse array to verify that the pattern programmed into the device is correct. However, verification does not guarantee functionality. Devices can pass array verification but fail in the circuit board. These are called post-programming functional rejects. Post-programming functional testing simulates the actual operation of the device to verify functionality. This testing detects these functional rejects before they get into your system. The typical post-programming functional reject rate for PAL devices is about 0.5–1.0%, or 5000–10,000 PPM. Our AC-functional testing options for ProPAL, HAL and ZHAL devices offer 50 PPM quality levels.

## ProPAL, HAL and ZHAL Device Functional Testing

Thorough functional testing is at the heart of our ProPAL, HAL and ZHAL device program. We offer a range of programming and testing options. These are discussed in detail in this section.

All functional testing of PAL devices starts with test vectors, but the similarity ends there, because when it comes to functional testing of PAL devices, all vectors are not alike. They range in complexity from simple, short pseudo-random testing sequences (e.g. Data I/O's Fingerprint™ Test) to sophisticated structured vectors generated by expensive software programs used interactively by dedicated test engineers.

We offer a sophisticated brand of signature analysis testing and two types of structured vector testing. The signature analysis testing is performed on the AutoVec tester. The two structured vector testing options are straight-functional vector testing and AC-functional vector testing. Straight-functional testing provides full functional and DC threshold testing. AC-functional testing includes this straight-functional testing plus functional testing for AC conditions. The following is a brief description of these testing options and what they yield in final quality.

### AutoVec Testing

The AutoVec tester is an extremely sophisticated signature analysis tester. It generates a sequence of up to 20 million vectors via a proprietary hardware-based pseudo-random vector generation algorithm. These vectors provide a typical quality level of 500–700 PPM for finished product.

AutoVec testing requires less engineering interaction and setup time than straight-functional and AC-functional testing (while still providing excellent quality levels), and the business conditions for ProPAL devices produced with AutoVec testing are more flexible. These more flexible business conditions include lowered NREs and reduced minimum volume requirements.

### Straight-Functional Testing

The straight-functional and AC-functional testing options offer the next level of quality in the ProPAL, HAL and ZHAL device program. The increase in quality is a result of the increasingly sophisticated structured test vectors and test equipment used for testing.

When generating vectors for this level of testing we start with a transistor-level schematic of the device under test. With this we can model internal gate-level stuck-at-faults. Most other vector generation packages use inexact models—such as logic diagram representations—as the basis for vector generation. With our exact device representation it is possible to cover all possible faults of the internal gates (where these faults can occur).

Monolithic Memories test engineers, using proprietary software, then check for three-state faulting. This is in addition to the stuck-at 1 or stuck-at 0 faulting tested by most commercially available test packages and conventional testing methods.

Monolithic Memories proprietary software is then used to check the design for potential design problems (e.g. race conditions). Additional vectors are added as needed to test for these conditions and guarantee reproducible test results.

In the process Monolithic Memories test engineers interactively add vectors until 100% of the detectable faults are covered. The end result: guaranteed 90% fault coverage on every pattern we test, with typical fault coverage >95% and typical system quality levels of 200–400 PPM. If 90% fault coverage cannot be obtained our Field Application Engineers will work with the customer to improve the testability of the design, or we will continue processing your product upon receipt of a signed waiver.

### AC-Functional Testing

The starting point for AC-functional testing is the straight-functional DC vector set. The excellent functional testing coverage the straight functional vectors provide is now extended to threshold condition AC testing. Monolithic Memories test engineers, working with additional proprietary in-house "intelligent" software, use these vectors as the basis for generating their AC test vectors. For high-quality AC testing, the design must be considered. Multiple feedback paths must be accounted for in the testing sequence. These "intelligent" software packages "learn" the design and flag the test engineer when special AC testing considerations are found. In this iterative process the engineer adds vectors to cover AC testing conditions. Typically, the number of DC functional vectors is doubled or tripled for full AC testing coverage. The expanded set of vectors is incorporated into a test program that performs DC, functional, threshold-functional and AC-functional testing, all at the VCC extremes.

The end result is excellent system level quality of < 50 PPM.

When you utilize our ProPAL, HAL and ZHAL devices program you are getting tremendous quality and manufacturing cost benefits—semicustom product without the risks:

- You can prototype your system and start production with standard PAL devices.
- The Non-Recurring Engineering (NRE) charges for ProPAL, HAL and ZHAL devices devices are far lower than those normally required for a semicustom circuit, and can be amortized over your first production quantity.
- You save on the cost of programming and testing devices. This also shortens your production cycle, since you can plug the devices into the socket with no additional processing.
- You save on the costs of generating test programs and functionally testing devices. All devices are fully functionally tested before they leave the factory.
- We provide you with custom marking. This saves you the added expense of stripping the mark from standard devices and remarking them with your own mark.
- You eliminate handling errors. When you use ProPAL, HAL and ZHAL devices, you are receiving board-ready product. No need to program, mark or test. And the elimination of these extra processing steps means the elimination of many handling steps, which can be the number one cause of component defects.
- You eliminate or reduce board and system-level reworking. The high quality levels provided by our ProPAL, HAL and ZHAL device offerings significantly reduces board reworking costs.

# Testability

## Introduction

With digital logic design, it is all too easy to design a circuit which merely implements a specified function. When production starts it is suddenly found that the circuit cannot be tested, or perhaps that tests cannot be performed economically. Dealing with this situation can, at the very least, have a negative impact on the introduction of the system into the marketplace.

Potential headache can be avoided by taking test issues into consideration during the initial design. Instead of just designing a circuit which implements a specified function, which is the bare minimum that must be accomplished, that function needs to be implemented in a manner which can be tested.

The purpose of this section is to establish the notion of testability and its importance, and then to provide ways of avoiding the most common untestable circuits. The issues will be discussed primarily in the context of logic design in PLD's, although they are also relevant for general logic design.

After testability has been discussed for general circuits, some specific testability circuitry on the PROSE device will be discussed. Finally, test vectors will be reviewed. Various kinds of vectors are mentioned, and the general tools available for vector generation will be summarized.

## Defining Testability — A Qualitative Look

A completely testable design is one in which any and all device faults can be systematically detected.

First note that the issue is one of devices, not designs. The design itself must work as specified; that is the main job of the design engineer. Once the design is implemented in a device, the issue is how to test the device to make sure that the design has been correctly implemented. Throughout this paper, then, it will be assumed that a particular design works as is; we will just be addressing its testability.

The easiest and most effective means of testing a circuit is through a systematic series of tests. A random set of tests may also do well, but does not yield much information regarding the testability of a circuit itself. No number of random (or systematic) vectors can test an inherently untestable circuit.

In order to be able to perform a systematic test sequence, every part of the circuit under test must be accessible, so that it can be controlled. Only then can each node be forced high or low as needed. This is essentially a requirement of complete *controllability* of the circuit.

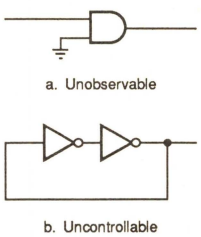
In order to be able to detect faults every part of the circuit must also be visible to the outside world, so that the results of each test can be observed. In this manner, each node can be inspected to determine its logic level. This requires complete *observability*.

These are, of course, the age-old issues of controllability and observability, which are as important for digital logic circuits as they are for so many other kinds of systems. If any portion of a circuit is uncontrollable or unobservable, then the testability of the entire circuit is compromised.

Figure 1 shows a couple of completely untestable circuits. The integrity of the top input in Figure 1a can never be verified. No matter whether it is shorted to ground, to $V_{cc}$, or whether it is functioning correctly, the output will be the same. That is to say, any faults on the top input cannot be observed at the output.

The circuit in Figure 1a would appear pretty useless as is. It is possible, however, that instead of being directly grounded, the second input may be driven by some distant signal, possibly on a different PC board, which happens to be a a logic low. If you cannot bring this line to a logic high, then it might as well be grounded.

The circuit in Figure 1b essentially has no input. This circuit can be thought of as a latch, but there is no way to change its logic state. Therefore, it is completely uncontrollable.



a. Unobservable

b. Uncontrollable

550 01

**Figure 1. Untestable Circuits**

## Quantifying Testability

In theory, if we want to quantify the testability of a given circuit, we might first attempt to make a list of all possible things that could go wrong with a circuit (no matter how unlikely), and then verify that all such "faults" can be tested, in all combinations and permutations. But for a circuit of any significance whatsoever, it will rapidly become apparent that this is not a practical solution.

What we need instead is a measure which can give an empirically reliable indication of the testability of a circuit, or of the quality of a given set of tests. There are several different such measures, but the most popular of these is the *single stuck-at faults* model.

There are several ways of analyzing circuits for single stuck-at faults. For very large circuits, various *testability analysis* schemes have been developed. However, for smaller circuits, especially of the size that would be put into a PLD, the more common method uses simulation.

### Simulating Single Stuck-At Faults

A given circuit is first simulated. The quality of the simulation is important; the more complete the simulation the better. A thorough simulation can then serve as a benchmark test sequence later. In this way, the fault simulation procedure also allows us to measure the quality of a given simulation, or set of tests, in addition to the testability of the circuit.

The results of the simulation are recorded. Next, one node in the circuit is modeled with a *"stuck-at" fault* — either *stuck-at-one* (SA1) or *stuck-at-zero* (SA0), as shown in Figure 2. The circuit is now resimulated. If the simulation results of the modified circuit are different from the simulation results of the good circuit, then the fault was detected. If not, then we have a faulty circuit which appears to operate correctly.

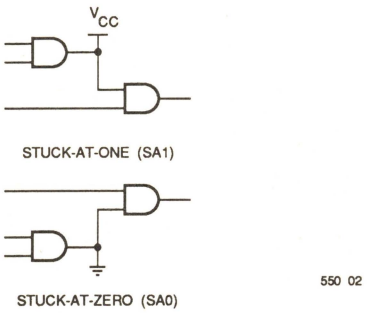



**Figure 2. Single "Stuck-At" Faults**

This procedure is repeated for each node, one node at a time (hence the name "single" stuck-at faults). The nodes are modeled with both SA1 and SA0 faults, so that for N nodes, we will have 2N simulations. If of those 2N simulations, D of them produced simulation results different from those of the original circuit, then we say that this simulation tested this circuit with a test coverage of D/(2N)*100%. Whereas this specifically tests only for single faults, experience shows that it is also a good test for multiple stuck-at faults.

### Undetected Faults

Why are some of the faults not detected? For simple combinatorial logic, there are two basic reasons: either the simulation was not complete enough to find the fault, or the circuit itself cannot be tested for the fault. So when an undetected fault is located, the first step taken is to add vectors to the simulations which will exercise the node being tested. By doing this, we gradually improve the quality of the simulation, and thus the quality of the test sequence that we can use in production.

It is possible that certain nodes will have undetectable faults for which no new vectors can be added. These are the result of an untestable design. It is the joint job of the test and design engineers to generate a test sequence that is as complete as possible. It is the design engineer's responsibility to provide a circuit which is testable. If both of these responsibilities are carried out, the result will be a testable circuit which can be tested with an exhaustive test sequence. This will yield the highest quality system. Note, however, that the overall responsibility is shared between the design and test engineers.

Needless to say, this process of analyzing the testability of a circuit is not done all by hand; software aids are used. There are many different kinds of programs that run on many different kinds of systems, ranging from PCs to workstations to mainframes. Some of them are standalone programs; others are integrated into larger overall environments. Their specific capabilities also vary, but in general, they can simulate a given circuit with a given set of vectors; analyze the test coverage that the vectors provide for the circuit; and generate new tests, either from scratch or by improving on the coverage of a few manually generated "seed" vectors. Most can also point out potential problems areas of a circuit, such as race conditions and logic hazards.

Finally, one frequently asked question is "So what if there is a fault that can never be detected. Who cares?" Theoretically, this question is not unreasonable. However, most companies will not feel comfortable telling a customer "We only tested half of the system, but if anything goes wrong with the other half, you'll never notice it." In addition, as will be seen, many untestable circuits occur as a result of poor design practices.

Testability issues for sequential circuits have implications far beyond the test bed. Indeed, failure to take these issues into account can greatly affect the normal performance of a system. The key for state machines is controllability. The challenge is to make all elements of the circuit controllable, both for testing and for general functionality.

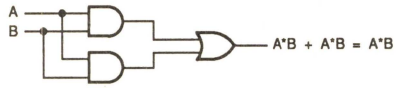
## Designing Testable Combinatorial Circuits

All of the previous procedures dealt mostly with the ways in which existing circuits are treated. However, if a finished circuit is found to be untestable, then it must be redesigned for testability. An easier approach is to design for testability from the beginning. Unfortunately there is no direct recipe for a testable design. There are, however, many common ways of making a circuit untestable. Most of this section is devoted to pointing out such problems.

The simplest kind of problem is *redundant logic*. Figure 3a shows one such circuit. It has a purely redundant product term. If the output of either of the product terms is stuck low, for any reason, then as long as the other product term is good, the fault will never be visible at the output.

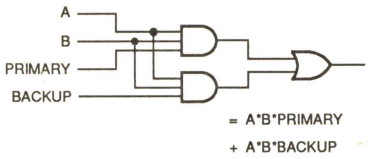
This may initially look like a benefit, since we have what we could call a "primary" circuit with a "backup." One can cover up some of the failures of the other (but not all failures). If this kind of

redundancy is truly desired, this is not the way to achieve it. When you ship out this circuit, you do not know if you really have a working primary and backup. The primary may already be malfunctioning; since it was never tested, you will never know. If you want useful, reliable redundancy, test circuitry must be added, as in Figure 3b, so that each part of the circuit can be independently tested.




550 03a

a. A Purely Redundant Circuit

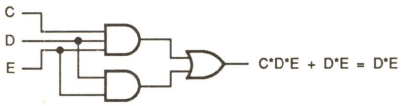



550 03b

b. Testable Redundant Circuit

**Figure 3. Making Redundancy Testable**

Figure 4 shows another redundant circuit. Although the product terms are not identical, the larger AND gate is really redundant. Any stuck-low faults at the output of this gate are not detectable.

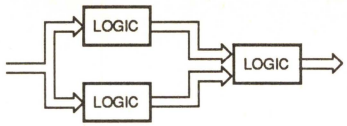



550 04

**Figure 4. Circuit with a Redundant 3-Input AND Gate**

## Reconvergent Fanout

Redundant logic is a special case of what is called *reconvergent fanout.* This is a term that refers to circuits that have inputs splitting up, going through independent logic paths, and then reconverging to form a single output, as shown in Figure 5. When this happens, it is very easy to introduce untestable nodes. It may not be easy to identify where such nodes are.

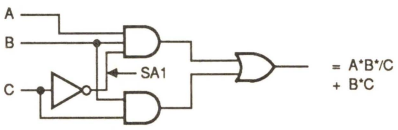



550 05

FANS OUT RECONVERGES

**Figure 5. Reconvergent Fanout**

Figure 6 is an example of a reconvergent circuit. The inputs are shared between two different product terms, which are eventually summed. This circuit appears harmless enough, but it turns out that the node indicated by "SA1" cannot be tested for a stuck-at-one condition. In other words, there is no way that we can guarantee that that node is operating correctly.



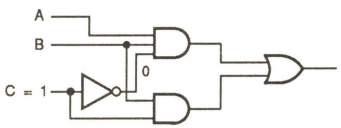

550 06

**Figure 6. A Reconvergent Circuit with an Untestable Node**

It is worth analyzing this circuit a bit more closely. This will give some insight into the kinds of analyses that are necessary when evaluating circuits and generating tests, and into the ways in which untestable nodes are created.

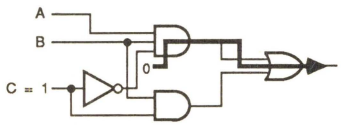
If we wish to prove that the node in question is not stuck high, then we must force it low and prove that we were successful in doing so. Thus we have two requirements: forcing the node low, and seeing the logic low on the output — controlling and observing the node.

First we raise input C high to force the node to a logic low condition, as in Figure 7a. This satisfies our controllability requirement. Next we need to provide a way to propagate this logic low to the output (Figure 7b). This is referred to as *sensitizing a path* to the output. The first step is to get the logic low past the AND gate. But if either input A or B is low, then the output of the AND gate will be low regardless of the node being tested. Thus we must force both A and B to a logic high, so that if there is a low on the output of the AND gate, we will know for sure that it came from the node we are testing. This is shown in Figure 7c.



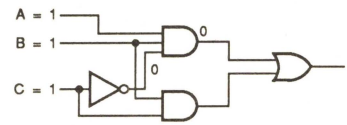

550 07a

a. Controllability: Forcing the Node Low



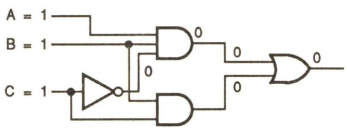

550 07b

b. Observability: Sensitizing a Path to the Output




550 07c

c. Propagating Past the AND Gate




550 07d

d. Propagating Past the OR Gate Sets Up an Impossible Condition

**Figure 7. Analyzing Testability**

Next we wish to get the logic low through the OR gate to the output. To do this, we must insure that the second OR input is always low; if it is high, then the output of the OR gate will be high regardless of the node being tested. If we can keep the lower OR input low, then if the node we are testing was sucessfully forced into a low condition, then the output will be low. Otherwise the output will be high. This can be seen in Figure 7d.

How do we keep the lower OR input low? By making the output of the lower AND gate low, which can be done by setting one of its inputs low. However, we have already required that all of the inputs be high. Thus we have required a set of conditions that cannot be met. One of three things will result:

1. The lower AND gate has both inputs high, and therefore keeps the lower OR input high. In this case, we may have been successful in forcing the node under test low, but we cannot see it at the output.

2. We bring input B low, allowing the lower OR input to go low. However, now the output of the upper AND gate will always be low. So we will see a low at the output, but we cannot be sure exactly where the low came from.

3. We bring input C low, allowing the lower OR input to go low. However, now we are no longer forcing the node under test low.

So we can either force the node low, but cannot see the low at the output; or, we can see a low at the output but cannot be sure of its source; or, we cannot force the node itself low. In any case, we will never be able to guarantee that the node under test is not stuck high.

Note that the two "independent logic blocks" which generate the signals that eventually reconverge are testable by themselves; they are just AND gates. It is only when we hook them together via the OR gate that the overall circuit becomes untestable. Thus *the testability of individual portions of a circuit does not guarantee that the entire circuit will be testable when the testable pieces are all connected.*

We can minimize this circuit using the following steps:

$$\begin{aligned} A*B*\overline{C} + B*C &= A*B*\overline{C} + B*C + A*B*B \text{ (by consensus)} \\ &= A*B*\overline{C} + B*C + A*B \\ &= A*B + B*C \end{aligned}$$

Thus the node we were trying to test is really not needed in the logic. The resultant circuit is shown in Figure 8, and is completely testable.

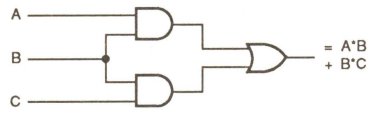



**Figure 8. The Minimized Circuit is Testable**

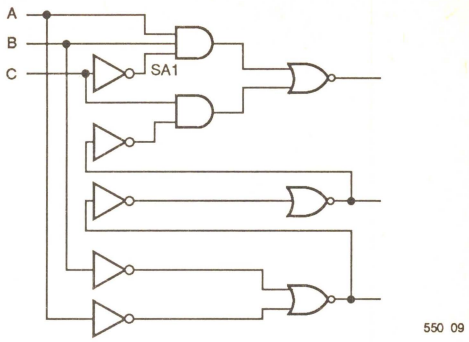



**Figure 9. A Messy Reconvergent Circuit**

Not all reconvergent circuits are so simple. Figure 9 shows a more complicated reconvergent circuit. Here some signals have to travel through several levels of logic to reach their final destination. This introduces considerable skew into the circuit, and will produce glitches on the outputs during certain transitions. In addition to this, there is again a stuck-at-one fault that cannot be tested.

Circuits like this can result from the design iteration process, as a designer tries to debug a circuit. By adding this and that, eventually the circuit works. But it is a mess, has poor timing characteristics, and is untestable. A little analysis of the logic itself shows that:

the bottom output is

$$\overline{(\overline{A} + \overline{B})} = A*B$$

thus the middle output is

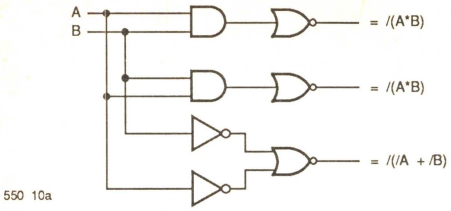
$$\overline{(A*B)} = \overline{A} + \overline{B}$$

which makes the top output

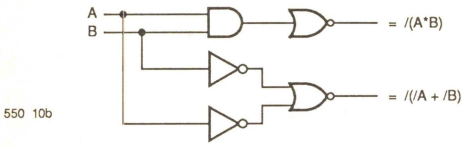
$$\begin{aligned} \overline{(A*B*\overline{C} + C*\overline{(\overline{A} + \overline{B})})} &= \overline{(A*B*\overline{C} + A*B*C)} \\ &= \overline{(A*B)} \\ &= \overline{A} + \overline{B} \end{aligned}$$

That is, the top two outputs are actually the same, and the third output is just the inverse of the top two. As convoluted as the original circuit looks, the logic itself is actually trivial. So if three outputs are really needed for some reason, we can generate them independently, as in Figure 10a. If only two outputs are needed, it is even easier. Figures 10b and 10c show two possibilities.

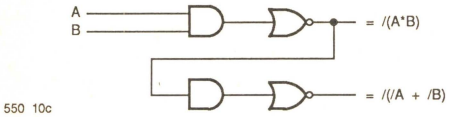
These circuits are much easier to understand, their timing characteristics are better, and they are completely testable.




a. A Cleaner 3-Output Version

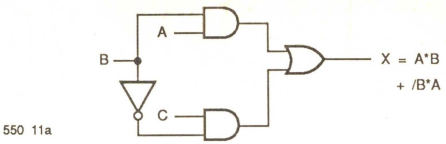



b. A Clean, Fast 2-Output Version

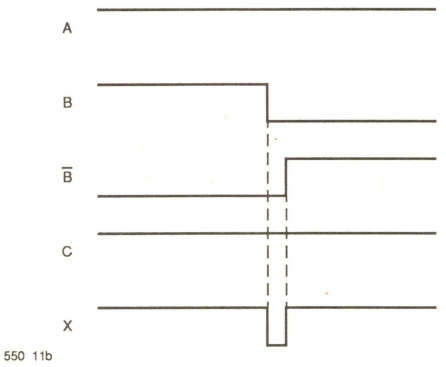



c. A Slower 2-Output Version.

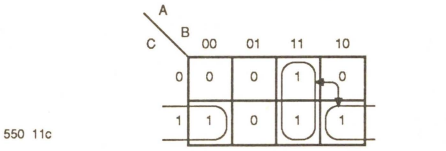
**Figure 10. Simplifying the Circuit of Figure 9.**




a. A Glitchy Circuit




b. Waveform for the Glitchy Circuit




c. "Gap" in the Karnaugh Map Indicates a Logic Hazard

**Figure 11. Examining a Glitchy Circuit**

## The Importance of Minimization

The common factor behind all of the untestable circuits we have examined is the fact that all of them were not minimal. By minimizing the logic, we made the circuits testable. This is true in general: *UNMINIMIZED LOGIC CANNOT BE FULLY TESTED.*

Very often, especially when designing with PLDs, an attempt is made to minimize logic only to the point where it fits into a particular PLD. Any further minimization is considered an academic waste of time. This is a grave misconception. Getting rid of all extra product terms, and eliminating all extra literals on the remaining product terms has real value. Failing to do so will result in untestable nodes in the circuit.

Minimizing is not always enjoyable, since hand techniques are usually too tedious, and Karnaugh maps are essentially useless for more than four or five inputs. However, computers have long been used to minimize logic. In particular, PALASM® software (version 2.22 and later) has a minimization routine which can minimize logic automatically before assembly.

## Logic Hazards

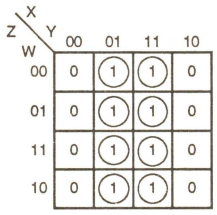
One occasional side effect of minimization can be the introduction of *glitches* into a circuit. Figure 11a shows such a "glitchy" circuit. The waveform in Figure 11b shows that under steady-state conditions, as long as inputs A and C are high, the output is high regardless of B. However, as B changes from high to low, causing the top product term to shut off and the bottom one to turn on, the inverter adds a bit of delay to the path that will turn on the lower product term. Thus the top term may shut off before the bottom one gets a chance to turn on. In this case, we have two logic low signals going into the OR gate, giving a low on the output. As soon as the lower product term turns on, the output goes back high, but not before the appearance of the high-low-high glitch.

Figure 11c shows the Karnaugh map for this circuit. It is minimal, but there are two product terms which do not overlap; they are "adjacent" in one location. These represent the two AND gates in the circuit diagram. The arrows indicate the troublesome transition: when A and C are high, and when B changes from high to low or the reverse. We can intuitively think of this as a "gap" between the two adjacent product terms, in which a glitch may occur.

Note that glitching is not a certainty. It is called a *hazard* because in certain situation, given certain timing situations, there is a chance that a glitch will occur.

Note also that the glitch is not really caused by the minimization process itself, but is caused by these "gaps" in the Karnaugh map. Unminimized logic with such gaps may also be glitchy.

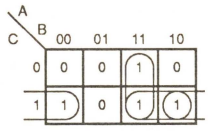
A PROM is a good example of such a circuit. PROMs can be used to implement any logic function of their inputs. However, regardless of the function, it is implemented in a completely unminimized fashion, using complete minterms. So even a function as simple as the one in Figure 12 (which could be implemented using a single product term, grouping all 1's into a single cell) is implemented with each 1 in its own cell. Thus there is a gap between every cell, meaning that every transition is a potential glitch. PROMs are notoriously glitchy, and it is for this reason that the output of a PROM is actually undefined until its access time has elapsed.




550 12

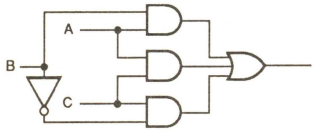
**Figure 12. In a PROM, Every Transition Can Glitch**

If we go back to the Karnaugh map in Figure 11c, we see that we can eliminate the gap — and the glitch — by adding a product term which overlaps both existing product terms and covers the gap. This is shown in Figure 13a, with the resultant circuit shown in Figure 13b.




a. A Redundant Product Term Can Eliminate the Glitch

550 13a



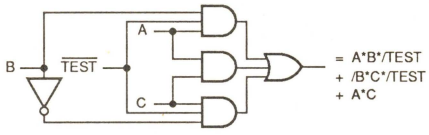

550 13b

b. A Glitch-Free, but Untestable Circuit

**Figure 13. Eliminating Glitches**

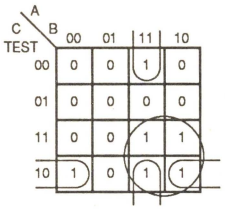
This circuit is no longer glitchy. Unfortunately, it is also no longer testable, since we have added in a redundant product term which cannot be tested (try it yourself). In order to have a circuit which is both testable and glitch-free, we must add a test input to the circuit which we can use to shut off the outside gates, isolating the middle gate for testing (Figure 14a). When the circuit is operating normally, the extra input is kept at a logic high condition, where it does not interfere with the basic logic function.

The Karnaugh map for this circuit is shown in Figure 14b. Note that all product terms overlap, but now the circuit is minimal. The size of the Karnaugh map has doubled, since we added another input. But if we isolate just that portion which corresponds to the test input being high, which is the normal operating mode (see Figure 14c), it looks exactly like the map of Figure 13a. Of course we should expect this, since we do not want the addition of a test circuit to affect the basic function.



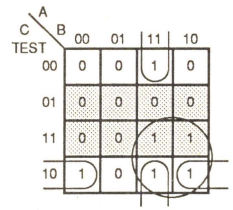

550 14a

a. A Testable, Glitch-Free Circuit




550 14b

b. Karnaugh Map




550 14c

c. Karnaugh Map Showing Non-Test-Mode Portion

**Figure 14. Making a Glitch-Free Circuit Testable**

Thus, in general, these types of glitches can be eliminated first by adding some redundant logic to get rid of the gaps in the Karnaugh map, and then by adding a test input to make the circuit testable.

## Designing Testable Sequential Circuits

The design of sequential circuits involves considerations above and beyond those required for simple combinatorial circuits. Latches and oscillators are circuits which appear combinatorial, but which use feedback to introduce sequential properties. State machines use flip-flops and feedback to generate what can be complex sequential circuits.

### Feedback

Whereas combinatorial circuits depend only on the conditions of present inputs, *sequential* circuits depend on both present conditions and past behavior to determine future behavior. This is made possible primarily by *feedback*. Feedback takes an output signal and routes it back for use as an input to the same circuit, as shown in Figure 15. We now have a situation where an output depends on itself; this can introduce new testability problems.

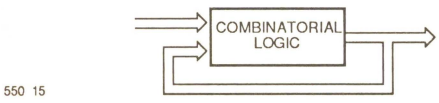



**Figure 15. Logic with Feedback**

Most sequential circuits (under varying circumstances also called *state machines*, *finite state machines*, and *sequencers*) make use of *flip-flops* as memory elements. These memory elements serve to remember a past condition (called a *state*) so that a future decision can be made based on it. This state is then fed back as in input. With PLDs, the flip-flops and combinatorial logic are contained within a single device, as shown in Figure 16.

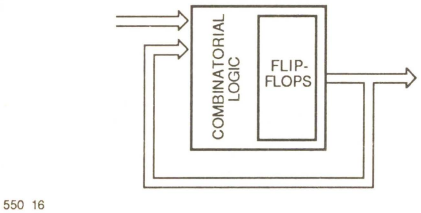



**Figure 16. Structure of a Sequential PLD**

Of course, the effects of feedback may have to be considered even when there are no flip-flops. The circuit in Figure 15 has feedback, but has no flip-flops. Such a circuit will either function as a *latch* or as an *oscillator*, as will be seen.

Before we look into the special needs of circuits with feedback, bear in mind that all of the testability criteria discussed for combinatorial logic still hold. The blocks of combinatorial logic shown in Figures 15 and 16 must be testable by themselves. What we will discuss here are issues which must be considered in addition to the issues involving combinatorial logic.

## Latches

A combinatorial logic circuit which uses positive feedback is a latch. The simplest possible latch is shown in Figure 17a. The output is fed back as an input in its TRUE form. This means, of course, that the output will stay at its present level; hence the name "latch".

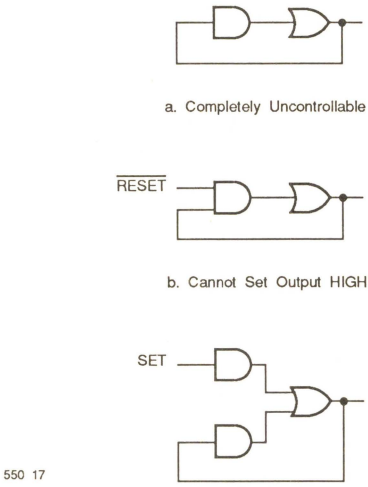



**Figure 17. Uncontrollable Latches**

The circuit as shown is clearly not useful, since it will always remain in its power-up state. If another input is added, as in Figure 17b, a HIGH output could be made to go LOW by setting the $\overline{\text{RESET}}$ input LOW. However, once the output goes LOW, there is no way to make it go HIGH again. Likewise, the circuit could be modified as in Figure 17c. Now a LOW output can be made HIGH by setting the SET input HIGH. However, once HIGH, the output can never be made to go back LOW.

### Controllable latches

For a latch to be useful, it must be completely controllable. The previous latches cannot be completely controlled. In order for a latch to be controllable, it must have both SET and RESET controls, as shown in Figure 18.

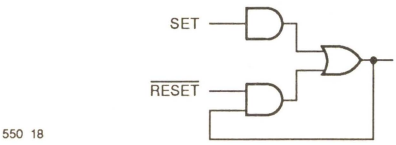
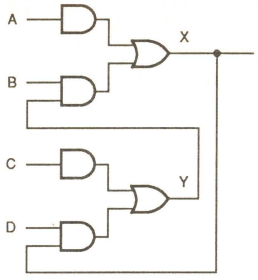


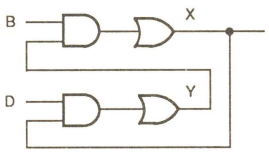

**Figure 18. A Controllable Latch**




X = A
+ B*Y
= A
+ B*(C + D*X)
= A
+ B*C ] SET
+ B*D*X
RESET

550 19a

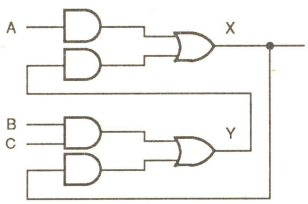
a. Latch with SET and RESET




X = B*Y
+ B*D*X
RESET

550 19b

b. Latch with RESET Only




X = A + Y
= A ] SET
+ B*C
+ X

550 19c

c. Latch with SET Only

**Figure 19. More Complex Latches**

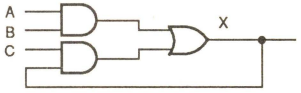
In PLDs, a latch can be detected by simplifying the logic for each function. If an output is a function of itself in TRUE form, then it is a latch. To be controllable,

- product terms containing the feedback should have at least one other direct input in the product (providing RESET control)
- there should be at least one product term with no feedback (providing SET control).

The circuit in Figure 19a provides an example. At first it is not immediately obvious that the circuit is a latch, but when the logic is simplified, we see that indeed it is. It is controllable since it has both SET and RESET controls. If the logic were shown in Figures 19b or 19c, the latch would be uncontrollable under some circumstances.

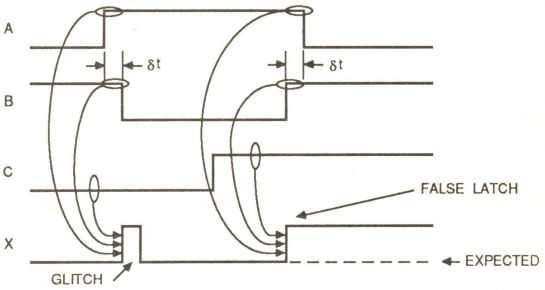
### Latch hazards

The circuit of Figure 18 can be generalized to have several inputs on both the set and reset controls. Such a circuit is shown in Figure 20. In this case, we have two inputs on the set AND gate. If the two set inputs A and B change from 0 and 1 to 1 and 0, respectively, then there will be a glitch or a false latch at the output if both inputs were 1 at sometime during the transition (Figure 20). For this transition, it is important to make sure that the 1-0 transition be made before the 0-1 transition to avoid anomalous output behavior. Merely delaying one input will not help, since it will delay both rising and falling transitions.




550 20a

a. Circuit



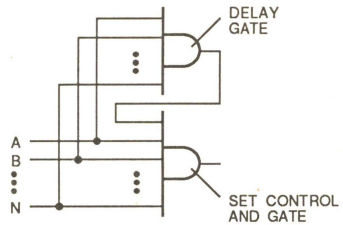

b. Glitch and False Latch

550 20b

**Figure 20. A Latch with More Complex SET Logic**

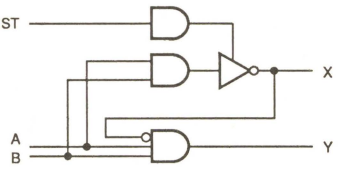
The simplest solution to this problem is the use of an edge-triggered flip-flop to synchronize the signals. This will eliminate any such glitches. If a flip-flop cannot be used, it is possible to delay reaction to a "11" condition to make sure that such a condition is not transitory. A circuit which accomplishes this is

shown in Figure 21a. This is relatively efficient in that only one delay circuit is required regardless of the number of inputs used on the set control (within the limits of the size of the AND gate). It will require an extra output on a PAL device.




550 21a

a. Circuit Which Delays "11...1" signals




550 21b

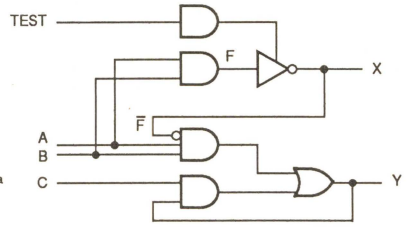
b. Testable Delay Circuit

**Figure 21. Delay Circuit**

This delay circuit will delay the effect of a "11" input by an extra propagation delay. However, it also provides a window of one propagation delay which will screen out any transitory "11" conditions that occur within that window. This allows up to one propagation delay's worth of skew between inputs during a transition from "01" to "10".

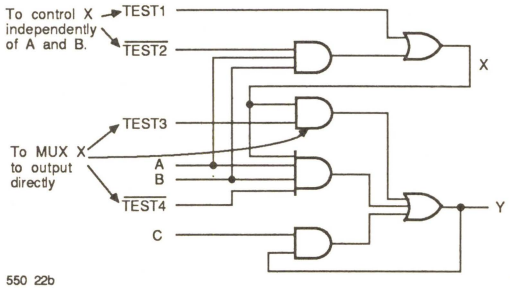
Because we have introduced redundancy, the circuit must be modified to be testable. If the circuit is implemented in a combinatorial PAL device, then programmable three-state can be used to test the circuit, as shown in figure 21b. By enabling output X, the redundant circuit can be observed without regard to Y. Then, to test Y, output X is disabled and then the pin is used as an input to drive the circuitry for Y directly. This provides a simple means of testing the circuit, but it only works if pin X can be measured and driven. The complete circuit is shown in figure 22a.

If node X is not so accessible, then additional circuitry and test inputs must be added. In the worst case, if node X is completely inaccessible, the resulting testable circuit is shown in figure 22b.



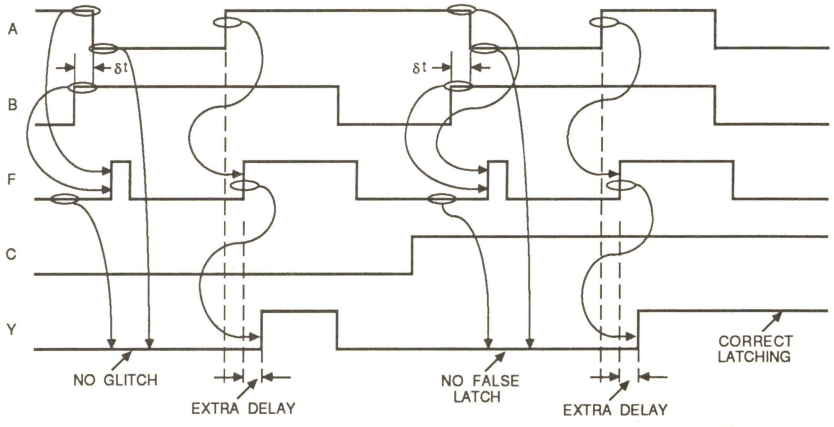

550 22a

a. Complete Latch Circuit




550 22b

b. Circuit if Node X is Completely Inaccessible




550 22c

c. Latch Circuit Behavior

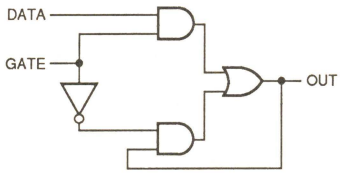
**Figure 22. A Testable Glitch-Free Latch**

Note that although the three-state capability is not needed, the circuit requires two extra gates, and, worst of all, four test inputs.

Figure 22c shows the behavior of either of the testable glitch-free latches.

### Transparent latches

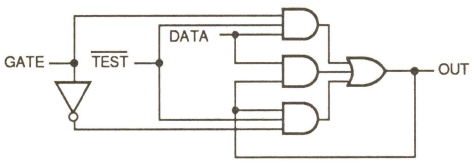
Many designers like to use PLDs to design standard D-type "transparent" latches. A D-type latch is a very simple circuit, shown in basic form in Figure 23a. As it turns out, however, this is a glitchy circuit of the type discussed on page 550-5 above. The problem is compounded in this case, since, given the right timing, the glitch can actually be latched; the glitching problem is no longer transitory. If this type of circuit is desired, it must be designed to be both glitch-free and testable; the resultant circuit is shown in Figure 23b.




OUT = GATE*DATA
+ /GATE*OUT

550 23a

a. Glitchy




OUT = GATE*DATA*/TEST
+ /GATE*OUT*/TEST
+ DATA*OUT

b. Glitch-Free and Testable 550 23b

**Figure 23. D-Type Transparent Latches**

## Oscillators

Circuits whose outputs are fed back in TRUE form are latches. If the outputs are fed back in COMPLEMENT form, then the circuit is an oscillator. A simple oscillator circuit is shown in Figure 24.

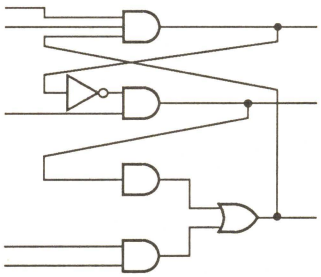


550 24

**Figure 24. A Simple Oscillator**

Latches are very often useful in circuits; oscillators rarely are. Crystals and other specialized oscillators are useful when it is necessary to generate a clock signal, for example. Trying to build an oscillator out of standard logic or PLDs will not yield a very predictable, accurate oscillator; where these circuits occur, it is usually by accident.

An oscillatory circuit may not always be obvious. It also may not oscillate all of the time. The oscillator shown in Figure 24 is uncontrollable; it always oscillates. However, just as we can design controllable latches, we can also design controllable oscillators (on purpose or by accident). This means that there may be an oscillator hidden in the circuit which will sometimes oscillate and sometimes be stable. Such a circuit is shown in Figure 25a.




a. Complete Circuit

X = A*B*D*E TERM 1
+ A*B*C*/X TERM 1

b. The Equation for X 550 25

**Figure 25. A Conditional Oscillator**

### Detecting oscillators

The oscillator in the circuit is not obvious. But if we simplify the logic completely, we can see that output X depends on /X; output Y depends on /Y; and output Z depends on /Z. Since the outputs are fed back to themselves in COMPLEMENT form, the circuit constitutes an oscillator.

This circuit will sometimes be stable. If we examine the logic function determining X, we see that it has two product terms, shown in Figure 25b. Term 1 is independent of /X; term 2 is dependent on /X. If inputs A, B, D, and E are all TRUE, then term 1 becomes TRUE, and the output stays HIGH regardless of the status of the rest of the circuit. It is thus stable. However, if signals D and/or E are LOW, then term 1 will be FALSE. If, at the same time, input C is HIGH, then, as long as the output X is LOW, term 2 will be TRUE, making the output HIGH (which makes the product term FALSE, which makes the output LOW, etc.). That is, the circuit oscillates.

In this manner, we can identify the conditions under which a conditional oscillator will oscillate. The mere presence of an oscillator is usually an indication that the circuit needs to be changed. It may be that the circuit only oscillates under conditions that could never possibly exist. One must be very certain of the impossibility of such a condition, however, if a conditional oscillator is to be tolerated. In addition, a thorough test sequence will usually expose a circuit to conditions that it may never encounter in a real system. Thus oscillators may interfere with the test process even if they do not disrupt the system.

## Designing Testable State Machines

State machines have their own set of controllability issues. These essentially boil down to the concepts of *initialization* and *illegal states*.

### State machine initialization

The nature of a state machine is that there is a well-defined sequence of states through which the machine will traverse as it operates. This implies the existence of a "first" state. Of course, these initial states vary from design to design. One obvious problem is the fact that many flip-flops — especially older varieties — do not power up in a predictable state.

### Power-up initialization

Flip-flops that truly power up into a random state must be initialized explicitly. Lately, however, flip-flops have become available which have "power-up reset". This allows the flip-flops to power up into a predictable state every time. This is helpful when the power-up state also happens to be the initial state. But even if it is not the initial state, a predictable initialization sequence can bring the state machine into its start-up state.

Unfortunately, such initialization schemes rely on the ability of the device to initialize itself when being powered up. If the system needs to be re-initialized, it will have to be completely turned off and then turned on again. Anyone who has had to turn off a computer in order to reboot will know that this is not an elegant way of re-initializing. By building initialization into the design, a means of performing a "warm boot" is provided. It is for this reason that initialization must be considered along with all other aspects of the design.

Some devices, such as the PAL32VX10/A, the PAL22RX8A, and other PAL devices, have mechanisms specifically designed for initializing a state machine. These are usually in the form of global set and reset product terms. By programming the conditions for initialization onto such terms, the device can be re-initialized at any time. Other devices, like the PMS14R21/A and the PLS devices, have pins which can be dedicated as preset pins.

### Including initialization in a design

Some of the simpler devices do not have specific provisions for initialization. However, the need is still present in these devices; here the initialization should be included in the design. This is a very simple process; it can be added in after all of the other design details have been worked out. Adding initialization will use up one input pin and potentially one product term on some outputs; this can affect the choice of device for the design.

To provide initialization in an otherwise complete design when Boolean equations are being used:

- determine the start-up state
- assign each bit as being initialized active or inactive, based on the desired start-up state
- if a bit is to be initialized inactive, add "/INIT" to every product term for that bit.
- if a bit is to be initialized active, add one product term consisting solely of "INIT"

Here we have assumed that the initialization pin has been called "INIT". "Active" would mean HIGH for an active high device; LOW for an active low device. "Inactive" is just the reverse.

The equation in Figure 26a can be initialized inactive as shown in Figure 26b, or active as shown in Figure 26c. Initialization is accomplished by asserting the INIT pin and clocking once. This "cookbook" approach is very reliable.

```
Q0 := Q1*Q2
    + Q2*/Q3
```

a. Uninitializable

```
Q0 := Q1*Q2*/INIT
    + Q2*/Q3*/INIT
```

b. Initialized Inactive

```
Q0 := Q1*Q2
    + Q2*/Q3
    + INIT
```

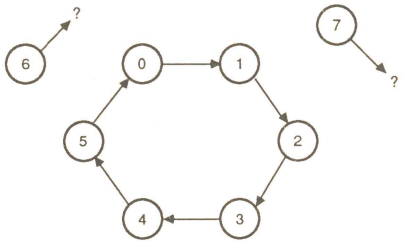
c. Initialized Active

**Figure 26. Designing in Initialization**

PALASM software also makes it possible to design state machines with a special syntax which essentially allows the state diagram to be transferred directly into a design file. For devices which have no dedicated initialization features, the initialization branches should be explicitly built into the state diagram. The software then performs the remainder of the processing needed.

### Illegal states

A state machine is formed by using a set of flip-flops to remember states, and assigning a code to each state. Since there are $2^n$ different codes that can be assigned to a group of n flip-flops, there is a good chance that some codes may not be used. For example, if a state machine is to have 6 states, 2 flip-flops will not be sufficient; 3 are needed. But 3 flip-flops allow 8 states, which will result in 2 unused states (see Figure 27).




550 27

**Figure 27. Illegal States**

Assuming that the state machine has been designed correctly, there is no reason why these extra states should ever be entered; therefore they are called "illegal" states. Unfortunately, situations do occur, thanks to noise and other unpredictable occurrences, which result in the state machine being in an illegal state. When this happens, the immediate need is to return to a normal sequence of states: *there must be a predictable means of getting from any illegal states into a legal state.*

Illegal state recovery is a controllability issue which actually affects functionality more than it affects testability. But the concepts used for functionality and testing are so closely related that it is worth treating here.

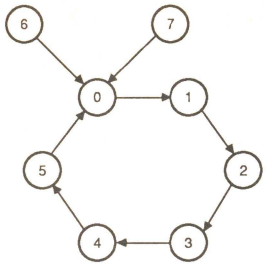
### Recovering from illegal states

There are three basic ways of getting out of an illegal state:

- re-initialize
- make sure that one can continue clocking until the machine recovers
- design the machine such that the start-up state is reached from any illegal state in one clock cycle, independent of any conditional inputs

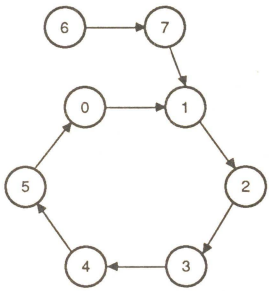
Of course, re-initializing will take the machine back into its start-up state from any state, legal or illegal (Figure 28). The disadvantage here is that outside control is needed to force initialization.

Very often, a path will exist which eventually takes the state machine back into a normal sequence (Figure 29). These paths are not usually designed in; they just happen to be there. In fact, if D-type flip-flops are used, it is surprisingly difficult to get a "closed" set of illegal states (that is, a set such that once one of the illegal states is entered, the machine will forever remain in




550 28

**Figure 28. Using Initialization to Recover**




550 29

**Figure 29. Cycling Back to a Legal State**

illegal states) by accident. In most cases, there will be a path which eventually leads back to a legal state. In these cases, merely clocking enough times will cause the machine to recover.

The drawback here is that one does not know ahead of time how many clock cycles will be needed. This necessitates some built-in way of knowing just when a legal state has been re-entered. And once that state has been reached, further cycling may be needed to get to a point where operation can resume.

### Designing-in one-step recovery

The most predictable way of dealing with illegal states is to provide a one-step path back to a legal state. Depending on the state desired, more or less work may be involved to do this. For PAL devices, we can consider three cases:

- all illegal states go to state 00...0
- all illegal states go to one state other than 00...0
- each illegal state goes to some legal state

The cause of poor illegal state recovery can be illustrated conceptually with Karnaugh maps (although realistically, Karnaugh maps are often not used). When calculating the equations for a particular bit, it is tempting to use Don't Care cells from the Karnaugh map (Figure 30) to simplify the logic. The success of illegal state recovery depends on how these Don't Care cells are treated.

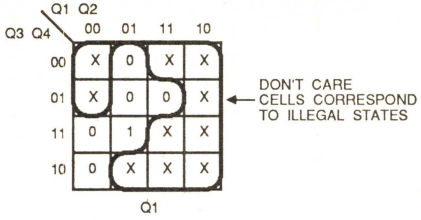
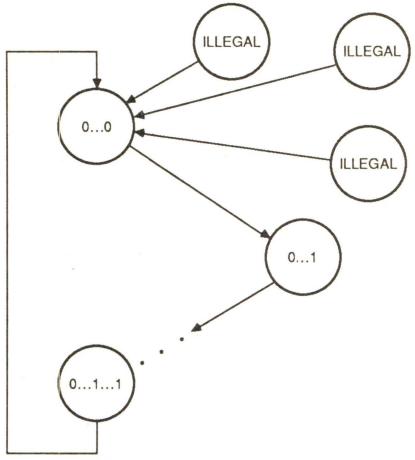


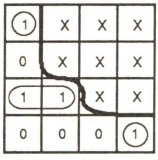

**Figure 30. Illegal State**

### Recovering into state 00...0

This is the simplest case; it is illustrated in Figure 31. It is accomplished by not using any illegal states to generate the logic for any of the bits. Since most PAL devices have only D-type flip-flops, a bit will go HIGH only as a result of legal states. Any illegal states will cause all bits to be LOW.




a. State Diagram



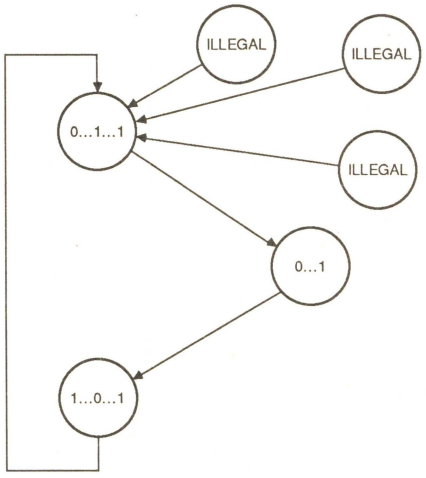

b. Karnaugh Map

**Figure 31. Recovering to State 0...0**

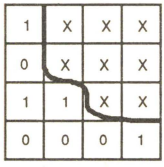
This procedure does not work when J-K or T-type flip-flops are used. In fact, it is deadly. Whereas a D-type flip-flop defaults to LOW, J-K and T-type flip-flops hold their present state as a default. Thus if illegal states are not considered in the transfer functions, an illegal state will cause the state machine to be locked up in that state.

### Recovering into one fixed state

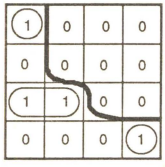
This case is shown in Figure 32a. The procedure can be illustrated conceptually with a Karnaugh map. It must first be decided which legal state will be entered, and the resultant value of each



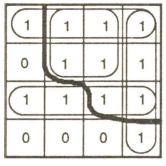

a. State Diagram




b. Karnaugh Map for Bit Qn




c. Bit Qn Recovers to 0




d. Bit Qn Recovers to 1

**Figure 32. Recovering to a State Other Than 0...0**

state bit. The Don't Care cells for each bit are then filled with the corresponding next state bit value; if the next state for a bit is to be 1, then Don't Care cells are filled with 1's for that bit's Karnaugh map; the procedure for a 0-bit is analogous. The equations are now taken by including either all Don't Care cells if filled with 1's, or none of them if filled with 0's. This procedure is illustrated in Figures 32b, c, and d.

When Karnaugh maps are not used, the same result can be obtained by explicitly considering all illegal states. When calculating the Boolean equations for:

- a bit that will be 0 after recovery, *no* illegal states should be included.
- a bit that will be 1 after recovery, *all* illegal states should be included.

When J-K flip-flops are used, then the transfer function for either J or K — but not both — will include all illegal states.

- If a bit is to be HIGH after recovery, *J* should account for all illegal states; *K* should account for none.
- If a bit is to be LOW after recovery, *K* should account for all illegal states; *J* should account for none.

This must be done explicitly for J-K flip-flops even if state 0...0 is the recovery state.

When T-type flip-flops are used, there is no easy way out; any recovery must be explicitly designed-in as part of the original function.

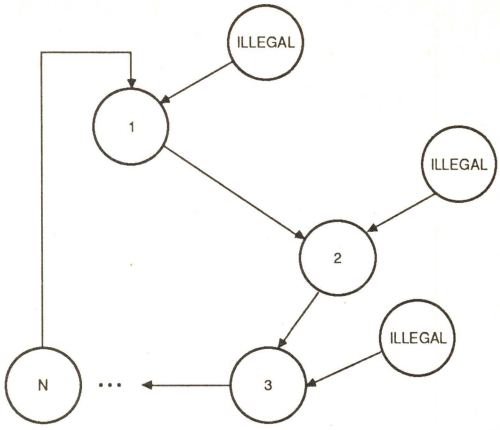
### Recovering Into Any Legal State

The third case allows one to fill in the Don't Care cells of a Karnaugh map in such a way that some legal next state is always reached in one clock cycle, but such that the 1's and 0's are placed to keep the logic functions simple. This is shown in Figure 33. The disadvantage here is that since different illegal states result in a different legal state, some additional cycling may be required to allow operation to resume.

When Karnaugh maps are not used, this can be implemented more simply by explicitly including the illegal states as part of the complete state diagram. This is especially simple if the state machine input format for PALASM software is being used.

### Default transitions

The PMS14R21/A and PLS devices have default branching mechanisms. When PALASM state machine input is used, it is possible to specify a DEFAULT BRANCH. This means that when in any state, if none of the branching conditions are satisfied, some user-definable state is automatically reached. This can be used as a way of recovering from illegal states.




a. State Diagram

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 |

550 33

b. Karnaugh Map

**Figure 33. Recovery Such That Logic Functions Are As Simple As Possible**

In PLS devices, the complement array can serve as a way of recovering from illegal states. In a design, only legal branches are defined. When in an illegal state, since no legal branch is active, the complement array is activated, allowing for some default state to be reached.

### Testing illegal state recovery

One of the difficulties of designing illegal state recovery into a circuit is the fact that it is difficult to test. Because the state is illegal, it is impossible to force the circuit into such a state. The use of register preload circumvents this problem. With preload, any state — legal or illegal — can be loaded into the register. If an illegal state is loaded, then the circuit can be tested to verify that correct recovery does indeed occur.

The use of preload must be considered carefully with devices having programmable asynchronous set and reset features. If these are driven by feedback from an output, then situations can occur where preloading one state immediately causes a set or reset to the opposite state (Figure 34). There are two alternatives: either avoid preloading such states, or include a control input in the set and/or reset product terms which can disable the feature when testing.

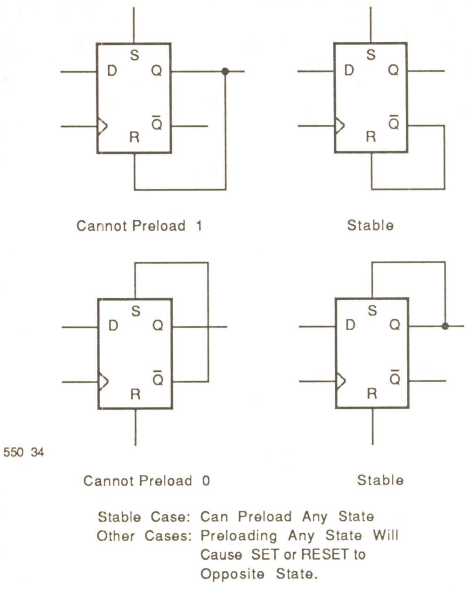



Figure 34. Preloading Registers with SET and RESET

### Providing for bed-of-nails testing

Most state machine PLDs are equipped with an enable pin for disabling the outputs. This is a key feature when the circuit board is to be tested in a bed-of-nails tester. When the devices driven by by the PLD are tested, it is recommended that the PLD be disabled so that there is no output level contention. Since the enable pin is usually grounded to keep outputs permanently enabled, it can instead be made available for use during testing.

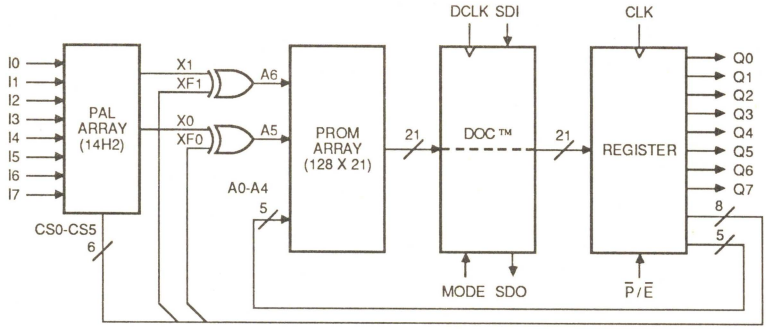
Note that for combinatorial devices, there is generally no output enable pin. The disabling feature is instead implemented through a product term. Designing the part such that the outputs can be disabled during bed-of-nails testing is also encouraged for these combinatorial designs.

## Designing for Testability With the PROSE™ Device

Today's more complex circuits and systems are becoming prohibitively expensive to test using standard methods. Diagnostics-On-Chip™, or DOC™, is a test feature provided in several of Monolithic Memories' devices as a means of increasing testability at the system, board, and chip levels. DOC is especially useful in the PROSE™ device (PMS14R21). It is even used by device programmers to configure the two programmable arrays inside the device (Figure 35).

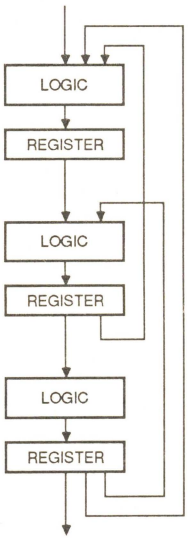
### DOC Architecture

Testability consists of two basic elements: controllability and observability. In a sequential (registered) system, these two elements are lost when a register is not directly accessible. In Figure 36a, the first register is not observable and the last register is not controllable. Figure 36b shows that the addition of a scan path through each register, as in the DOC method, provides the direct access for controllability and observability, which ensures complete testability.



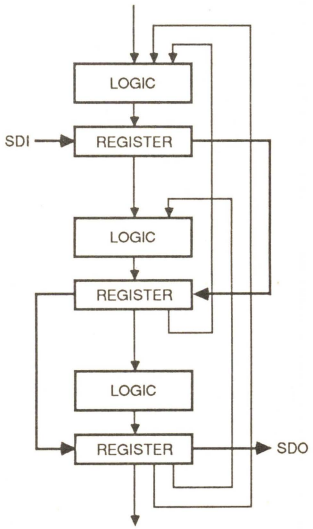

550 35

Figure 35. PROSE Block Diagram




550 36a

a. Standard Sequential Logic




550 36b

b. Sequential Logic with a Scan Path

Figure 36. Testability Can Be Increased by Providing Direct Access to All Registers

The heart of the DOC circuitry is the shadow register (see Figure 37). The shadow register is a serial/parallel register, equivalent in length to the pipeline register. It is called a shadow register because it is invisible to the device during normal operation. It is clocked by its own clock input, DCLK.

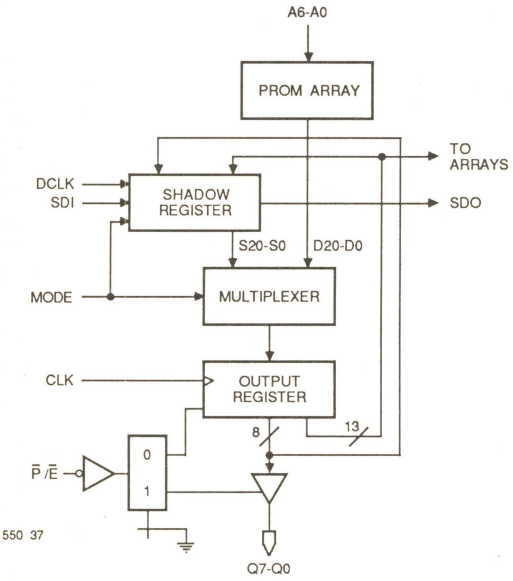



**Figure 37. DOC Circuitry in the PROSE Device**

In normal mode (MODE input is LOW), the shadow register operates as a serial shift register (see Figure 38). The Serial Data Input is SDI, and the Serial Data Output is SDO. The pipeline register can operate at the same time while MODE is LOW.

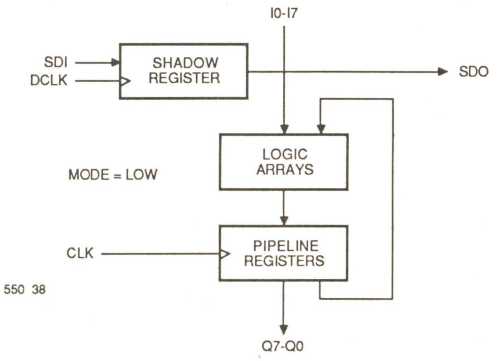



**Figure 38. The Shadow Register Operates as an Independent Shift Register when MODE is LOW**

In diagnostic mode (MODE is HIGH), the shadow register operates as a parallel register (see Figure 39). It can be parallel loaded from or to the pipeline register by clocking the receiving register. A swap can be performed by clocking both at the same time.

Note that the PROSE device differs from other DOC family members in that the shadow register is loadable only from the output register. Other devices also allow loading of the data on the output pins, if the device is connected to a bus. This does not affect the device-level testability if the outputs do not connect to a bus, of if the bus is otherwise observable.

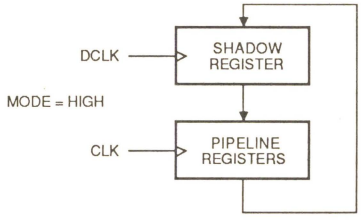



**Figure 39. The Shadow Register Can Parallel Transfer its Contents to and from the Pipeline Register when MODE is HIGH**

| INPUTS | | | | OUTPUTS | | | OPERATION |
|---|---|---|---|---|---|---|---|
| MODE | SDI | CLK | DCLK | Q20-Q0 | Q20-Q0 | SDO | |
| L | X | ↑ | * | Qn ← PROM | HOLD<br>Sn ← Sn-1 | S20 | Load output register from PROM array |
| L | X | * | ↑ | HOLD | S0 ← SDI | S20 | Shift shadow register data |
| L | X | ↑ | ↑ | Qn PROM | Sn ← Sn-1<br>S0 ← SDI | S20 | Load output register from PROM array while shifting shadow register data |
| H | X | ↑ | * | Qn ← Sn | HOLD | SDI | Load output register from shadow register |
| H | L | * | ↑ | HOLD | Sn ← Qn | SDI | Load shadow register from output register |
| H | L | ↑ | ↑ | Qn ← Sn | Sn ← Qn | SDI | Swap output and shadow registers |
| H | H | * | ↑ | HOLD | HOLD | SDI | No operation† |

* Clock must be steady or failling.
† Reserved operaton for 74S818 8-Bit Diagnostic Register.

**Figure 40. Diagnostics Function Table**

All of the functions of the DOC circuitry are described in the function table (Figure 40).

DOC allows access to all twenty-one pipeline flip-flops in the PROSE device through the serial path, requiring only four additional pins. These four pins can be controlled directly, or can be connected to other DOC circuits in series. A series connection of several DOC circuits allows the same four signals to address an unlimited number of flip-flops on a board or system (Figure 41).

A typical test would be performed as follows:

1. Test vector shifted into shadow register(s)
2. Test vector parallel transferred to pipeline register(s)
3. Device/system clocked desired number of times to run test
4. Test results parallel transferred to shadow register(s)
5. Test results shifted out of shadow register(s)

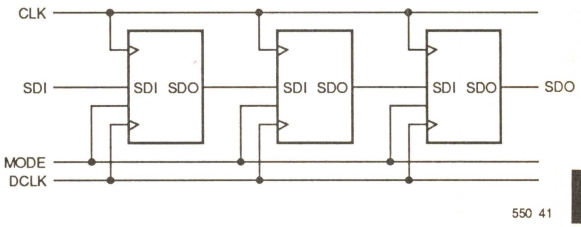



**Figure 41. Example Architecture for Use of System-Level Diagnostics**

Note that while shifting, the system can return to normal operation. In addition, while test results are being shifted out, a new test vector can be shifted in.

The swap function allows the state of the pipeline register(s) to be restored once the test is complete. When the test vector is parallel transferred to the pipeline register, the pipeline register contents are transferred to the shadow register at the same time for

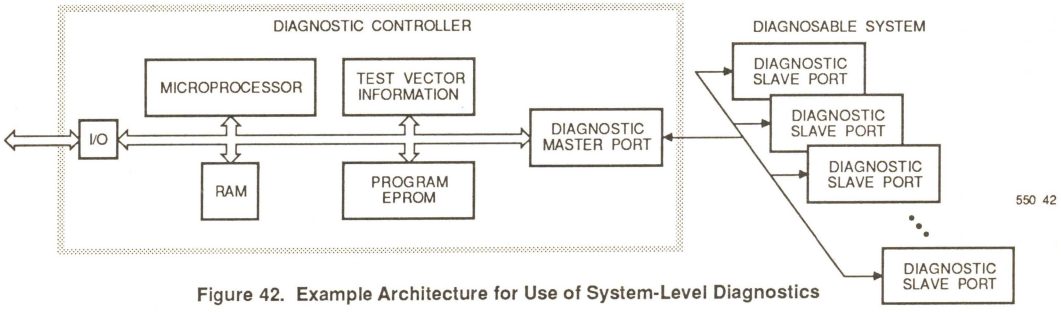



**Figure 42. Example Architecture for Use of System-Level Diagnostics**

storage. Later, when the test results are transferred to the shadow register, the original pipeline register information (stored in the shadow register) is transferred back to the pipeline register.

### System-Level Testing

At the system level, DOC provides the ability for a diagnostic controller to monitor the interior status of a system. The diagnostic controller could control several scan loops, selecting the loops required for the test needed (Figure 42).

However, many key products, such as microprocessors, are not available with the DOC function and cannot be part of the scan path. This limits the use of DOC for full system-level testing to selected manufacturers who can use this additional testability as an enhanceement to a larger system-level testability strategy.

In addition, little support is available for writing the test vectors that can be run through the DOC scan path. The software that is available is expensive and runs on large computers only. This is another factor that limits the use of DOC on a system level. On the board or chip levels, however, test vectors are much easier to generate and can even be found by running vectors through a known good unit.

A complete system-level test would require that most of the devices in the system shown in Figure 43 incorporate the DOC circuitry. Other devices in the DOC family are shown in Figure 43. Devices with circuitry equivalent to the DOC format are available from several other suppliers as well. Also, many gate array and standard cell manufacturers offer standard functions similar to the DOC scan path and can easily be included in custom designs.

| PART NUMBER | DESCRIPTION |
|---|---|
| PMS14R21/A | 128-state sequencer |
| Am29PL141 | 64-state sequencer |
| Am27S65/A | 4-K Diagnostic PROM |
| Am27S75/A | 8-K Diagnostic PROM |
| Am27S85/A | 16-K Diagnostic PROM |
| Am9151 | 4-K Diagnostic Static RAM |
| 74S818 | 8-bit register |
| Am29818 | 8-bit register |

**Figure 43. DOC Products Family**

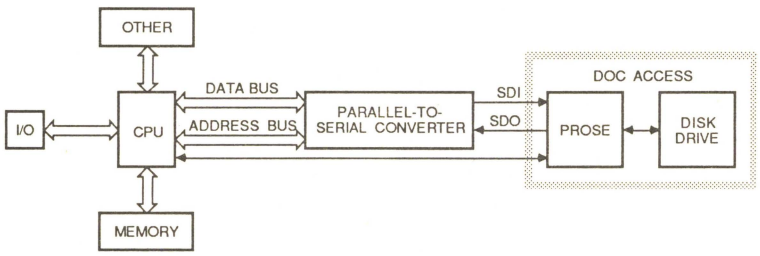
### Board-Level Testing

DOC in the PROSE device is especially useful at the board, or functional, level. The PROSE device will usually form the heart of a function, such as a peripheral controller. In addition, it often will serve to off-load the main Central Processing Unit (CPU) and be partially controlled by the CPU. DOC allows direct control of the PROSE device, bypassing a difficult-to-control CPU and taking command of whatever function the PROSE device performs (Figure 44). Here, on-board diagnostics can be easily done with the PROSE device. The alternative is to dedicate edge-connector signals to the DOC path.

The DOC circuitry provides access to the PROSE device's pipeline register. This can be used to set the outputs to a given state, in order to test the effect on the devices surrounding the PROSE device. Or, the pipeline register can be set to a given state and then left to run freely, to verify functionality. If combined with control of the device inputs, the sequencer can be stepped through a number of states, to test the response of the surrounding logic. This is especially useful for bed-of-nails board-level testing; the PROSE device can be tested completely without having to be backdriven.

### Device-Level Testing

On the device level, the DOC circuitry effectively provides a Preload function for the register. Instead of loading the register from the outputs, as with standard PAL® devices, the register is preloaded from the shadow register. A standard type of preload is not possible on the PROSE device because thirteen of the flip-flops are buried (Figure 45). Preload is necessary for testing the device functionality, since the buried flip-flops must be set to a known condition before the device can be tested.




550 44

**Figure 44. DOC Allows Direct Access to Peripheral Elements In a System, Bypassing the CPU**

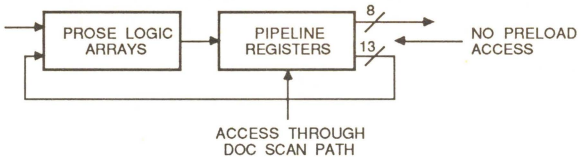



**Figure 45. DOC Effectively Adds a Preload Function to Buried Flip-Flops**

The DOC circuitry allows more than just a Preload equivalent, however. It also allows observation of the pipeline register, which contains all of the state information. Thus, an individual state transition may be tested by preloading the desired state, setting the inputs, clocking the device, and then observing the resulting state in the pipeline register. State transitions which do not result in a change in outputs are thus easily tested (Figure 46).

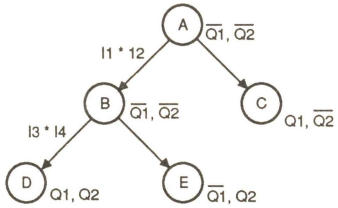



**Figure 46. DOC Allows Transitions that Do Not Result in Changes to the Outputs to Be Verified. In this Case, the Transition from A to B Does Not Change the Outputs (Q1 and Q2), but the Internal Feedback Changes (the Condition Select Signals Examine I3 and I4 Instead of I1 and I2). Buried Flip-Flop Observability Is Required to Verify the Transition**

## Using Test Vectors

Digital systems are generally tested by applying a sequence of test vectors. A test vector is a group of signals which are applied (forced) and measured (sensed) on a device or a board. The vector thus defines all inputs and expected outputs for a given test. As we have noted, the sequence of tests performed greatly affects the quality of the overall tests, as measured by the fault coverage.

In general, we can talk in terms of three kinds of vectors. *Simulation* (or application) vectors, *functional* test vectors and *signature* test vectors.

Simulation vectors are generated during the design process. Their main purpose is to help the designer verify that the design has been correctly implemented. They represent the way in which the circuit was intended to operate. When PALASM software (or almost any other PLD design software package) is used, simulation may be performed prior to programming a device. The software simulates the operation of the circuit, and then generates vectors from the simulation, adding the vectors to the JEDEC file. These vectors can then be used for testing by programmers that have the capability of performing functional tests.

While simulation vectors may be adequate for verifying that the design is operating as expected, they generally do not provide very extensive test coverage. For this reason, we distinguish functional test vectors from simulation vectors.

It is very difficult to generate a complete set of functional test vectors by hand; computer programs are generally used instead. The simulation vectors are often used as a basis for generating a more comprehensive set of functional test vectors; in this capacity, the simulation vectors serve as *seed* vectors. There are many programs which perform this function although many of the programs require larger computers and take a long time to run. Monolithic Memories also generates functional test vectors for patterns that are used in ProPAL and HAL devices. This is discussed more fully on page 3-106.

More recently, programs which run on the IBM PC-compatible computers have been developed to generate vectors for use in testing PLDs. Most well-known among these are PLDtest™ from Data I/O Corp., and TestPLA™ from Structured Design. These programs use the programming information in the JEDEC file to generate tests.

On most patterns, they can generate test sequences of high quality. If complex internal feedback is used in a particular design, then some manual test generation may still be needed to improve the test coverage. Both of these programs support the use of register preload for initializing states; the TestPLA package can also generate tests for devices which do not have the preload feature.

While functional vectors provide more extensive tests, they may not exercise the circuit in the manner in which it was meant to be used. Thus, for example, a conditional oscillator in a circuit (as discussed above) may not be a problem during simulation, since the conditions causing oscillation are not thought to be possible by the designer. However, the functional vectors will take all situations (some of which may not be physically possible) into account in the tests. Thus more subtle design problems may become apparent when functional test vectors are generated.

Signature vectors are random vectors which are first applied to a device which is known to be good in order to generate a "signature". This same set of vectors is then applied to a device of unknown quality; if the same signature results, the device is said to be good; if a different signature results, then the device is assumed to be faulty.

Signature vectors can vary greatly in the quality of testing they can provide. Since they are generated with no knowledge of the circuit being tested, many more vectors must be used to perform a good test. The quality of the test depends on the circuit being tested, the number of vectors used, the speed with which the tests are applied, and the algorithm used to generate the vectors. The tester must also be able to apply a preload sequence to devices that have registers; otherwise two devices may power up into two different states. In that case, both devices will generate different signatures even if both are good devices.

Quality signature testing can be very cost effective, since no advance knowledge of a device pattern is needed. This reduces the amount of resources that must be dedicated to test vector generation. Signature testing options are discussed more fully on page 3-106.

The different types of vectors are summarized in Table 1 below.

| TYPE OF VECTOR | PURPOSE | GENERATED BY: |
|---|---|---|
| Simulation (Application) | Used for verifying whether or not a design will operate as expected when implemented. | Sequence defined by the design engineer, usually by hand. Actual vectors generated by design software, placed in the JEDEC file. |
| Functional | Used for verifying that a device is operating correctly. | Usually generated by a computer program such as PLDtest or TestPLA. The simulation vectors can be used as seed vectors |
| Signature | Used for verifying that a device is operating correctly without functional vectors. | The tester generates the test sequence during the test. |

**Table 1. Test vectors**

## Summary

The time to start considering ways of testing a circuit is before the circuit has been designed. The key to testability lies in the way the circuit is implemented.

Basic combinatorial logic can be made completely testable simply by minimizing logic. It is not even necessary to analyze the circuit for redundancy or reconvergent fanout; automatically minimizing all logic will eliminate any occurrences.

Where a sequential circuit is generated from simple feedback paths in the logic, the circuit must be analyzed as a combinatorial circuit. All combinatorial logic must be included to determine whether the circuit is a latch or an oscillator. If a latch is desired, it should be completely controllable. If an oscillator is found, it is probably not desired, and will generally indicate a mistake in the design. If a conditional oscillator is to be tolerated, one must be sure that the oscillation conditions can never occur, and that the test procedure will not cause oscillation.

In general, combinatorial circuits should be analyzed completely for the presence of latches and oscillators (wanted or unwanted). This can be done by simplifying each combinatorial logic block to see whether any signal ultimately depends on itself.

When the sequential nature of a circuit is derived through the use of flip-flops to generate a state machine, the two key issues are initialization and illegal state recovery. A combination of device features and careful circuit design will yield circuits that can behave predictably even in unexpected situations.

DOC is a testability feature that is useful in the PROSE™ device; it may be used on multiple levels: system, board, and chip. While the system-level uses may be restricted by the limited availability of support products, the board or functional-level uses are exceptionally handy when the PROSE device acts as a local controller. And on the device level, the DOC circuitry provides a means of accessing the buried flip-flops within the device for functional testing.

It is important to analyze the testability of a circuit before committing it too far. Thus any changes can be made early on. In particular, if the test analysis software points out any logic hazards in your circuit, you can easily remedy them by modifying the design.

These simple steps, taken early in the design phase, can help avoid later redesigns, and ultimately provide a higher quality system.

Finally, the ultimate test quality depends also on the quality of the test sequence used for production, functional test vectors and high quality signature tests will provide you with the highest confidence in the quality of your system.

# MONOX™ 3
# Oxide-Isolated Process

Monolithic Memories' premier programmable logic process, MONOX 3, is an evolution of the junction-isolated process used in the popular 15 nanosecond PAL family. The 15 ns PAL device process is a shallow-junction, ion-implanted, diffused isolation technology.

When the time came to advance PAL device speed through improved process technology, the decision was made to evolve from and benefit from the proven reliability, simplicity, and manufacturability of the 15 ns PAL device process. Only fully recessed oxide isolation and stepper design rules were to be added for the new technology. The fully recessed oxide isolation technology to be used had already been proven in earlier processes.

## MONOX 3 Process Description

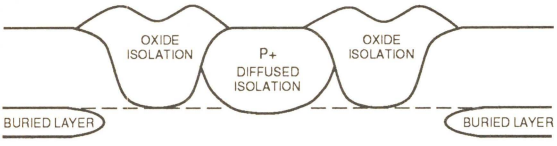
The unique feature of MONOX 3 is the isolation structure (patent pending) which combines the best features of fully recessed oxide isolation (FULROX) and diffused isolation, while maintaining a very dense structure. The advantages of FULROX, low capacitance and high density, are well known to the industry.

Diffused isolation has an important advantage for transistors that are driven hard into saturation, as in the case where minimum size array transistors are used to program fuses in PAL devices. In this case, substantial current is injected into the substrate, and this may adversely affect nearby circuitry. While this substrate injection can be reduced in FULROX, it has an adverse effect on capacitance and perhaps density.

In MONOX 3, the diffused portion of the isolation acts as an excellent substrate contact and as a sink for the injected substrate current. This permits the FULROX to be optimized both for density and for low capacitance, lower than is typical for industry-standard oxide isolation. The typical density disadvantages of diffused isolation are minimized by containing the diffusion within the FULROX. This isolation structure results in a die that is substantially smaller than some trench-isolated products, and that has lower capacitance than other oxide-isolated products.

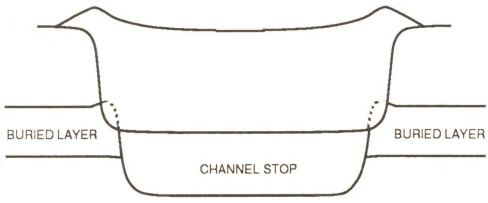
Other features of the MONOX 3 process are:

- Fully ion implanted except for buried layer—This permits excellent control of the layers for a consistent product, and permits a base width of 2000Å which yields a cutoff frequency $f_t$ of 4.3 gigaHertz.
- High-pressure oxidation—This is used for the recessed isolation to minimize process temperature and crystal defects.
- Oxide walling all devices—This eliminates potential leakage paths that might cause reliability problems.
- Planarization of the isolation "bird's head" shape—This improves lithography and metal step coverage.
- N+ and P+ sink diffusions—These lower parasitic resistances.
- Dry (plasma) etch—This improves control and density of most layers including metal.
- Platinum silicide Schottky diodes—These prevent saturation of the logic transistors for improved speed.
- Titanium Tungsten fuses—These are simple and reliable.
- Double layer metal, with intermetal planarization—The first metal pitch is 4.5 microns and the second layer metal pitch is 6.0 microns.
- Stepper lithography with 1.5 micron minimum design rules (1.3 micron fuses)—This not only makes the die more compact, but significantly improves the fuse programming (see next section).




422 01

**MONOX 3 Isolation**




422 02

**Typical Oxide Isolation**

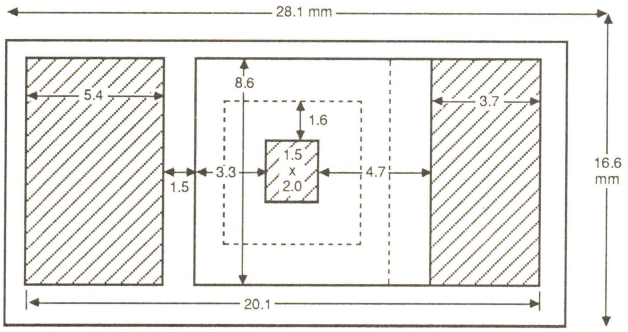
## MONOX 3 Fuse Technology

The fuse technology in MONOX 3 is Titanium Tungsten (TiW). This fuse technology has been used for years in millions of chips that have proven to be the industry's most reliable programmable logic parts.

In MONOX 3 the fuses are further enhanced by using stepper lithography to print them 1.3 microns wide. This significantly lowers the programming current from 70 milliamps to 35 milliamps maximum. A lower programming current means less power and heat are needed, leading to increased reliability and a denser chip design.

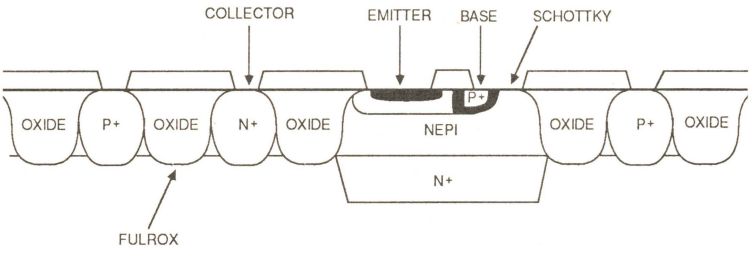
## MONOX 3 Summary

In conclusion, MONOX 3 was designed to be and is both high-performance and simple. Only thirteen masking layers, two diffusion cycles, and four oxidation cycles are used. This yields a process that competes with, and out-performs, other currently available programmable logic technologies. The relatively few steps needed to manufacture MONOX 3 devices mean fewer potential problems and increased reliability.




422 03

MONOX 3




422 04

NPN Transistor

# Product Assurance

## Introduction

Product Assurance consists of Quality Control, Reliability Assurance, Quality Assurance for Military Products and Quality Assurance for Commercial Products.

## Quality Assurance

Quality Assurance for the Commercial Products includes customer support and failure analysis, outgoing inspection and factory support, document control and quality information.

In order to support customers, QA provides actual data generated during various monitors and inspections throughout the factory. Non-proprietary data is available upon request. If the specific data is not immediately available, experiments can be run to collect the necessary information subject to resource and time limitations.

Another aspect of customer support is the performance of failure analyses. Failure analyses are broken down into three levels. Level 1 analysis consists of failure verification using an automatic tester. The result is only whether or not the device under test is good or bad with a datalog pointing out the potential failure mode. Level 2 analysis consists of Level 1 plus the verification at a bench top setup which results in confirmation of the failure mode with detailed specific data. Level 3 adds to Level 2 a decap and physical analysis to isolate the failure mechanism. Monolithic Memories has the capability to perform the total analysis in house, but occasionally sends analyses to outside sources whenever circumstances deem it necessary. Quality Engineering goals for cycle time are to complete Level 1 analyses within seven days, Level 2 within 14 days, and Level 3 within 30 days.

Not all customers want or need a full Level 3 analysis every time. The level can be specified at the time of submission. A failure analysis should be requested through the Sales person or Field Applications Engineer (FAE). A failure analysis request form will be completed at that time and the request form and suspected failure(s) will be forwarded to Quality Assurance.

Upon completion of the analysis, a formal report will be made to the requester. The report will summarize the results of the analysis and enumerate the steps followed in performing the analysis. In most cases, a statement of the corrective action taken to prevent future occurrences will be included for valid failures. In those cases where no failing condition is found, the device(s) and report are forwarded to the requester.

Quality Assurance supports the factory with periodic auditing of the various processes, areas and products. The Discrepant Material Reporting system (DMR) provides the factory feedback on the level of quality it is producing as well as providing protection to our customers via a gating of the product. The Quality Inspection group takes a 200 piece sample (.065% AQL for lot sizes according to Mil–HDBK–105D) from each production lot for visual and mechanical testing and electrical testing. The lot is returned to production for rescreening if a defective unit is found in the sample. The results of the inspections are summarized and reported weekly. Through programs aimed at solving the causes of the defects, Monolithic Memories has improved quality levels significantly.

As a customer, you can learn from our experience. Our data suggests that handling is the number one cause of defective material. Whether it is human handling or not, the product flows should be engineered so as to minimize the number of separate handling steps. By ordering completed product in even box quantities, no handling should be necessary after the product has been packaged by the factory. Product can be placed into boards upon receipt.

By placing product directly into boards without incoming inspection and handling, ship-to-stock has been accomplished. Ship-to-stock, also known as dock-to-stock or certification, is an electronics industry goal. It accomplishes minimum cost objectives of our customers. The ship-to-stock decision is a customer decision that is based on the confidence one has in their supplier to provide consistent, high quality. Monolithic Memories has mechanisms in place to support ship-to-stock programs and has a generic recipe for the certification process for those who would like to get a head start on a such a program. Refer to the outline of Guidelines for Product Certification.

## Quality Improvement Programs

The Product Quality Objective is divided into the following components:

- Reliability
- Electrical Performance
- Programmability
- External Visual/Mechanical
- Internal Die Visual
- Total Quality Process
- Quality Partnership Program

### Reliability

The reliability program consists of new product, package and process qualifications, qualifications of product, package and process changes, and product, package, and process monitors. Most reliability testing is performed on site in Santa Clara.

Qualification requirements are generally customer driven. Monolithic Memories has developed a procedural specification that attempts to cover most of the requirements of our numerous customers. Most test methods follow the Mil-Std 883 method if applicable.

The most common stresses are operating life, temperature cycling, and temperature and humidity testing. Monolithic Memories performs operating life at 125 degrees Celsius, 5.25V Vcc, and 1000 hours per method 1005, condition D. In some instances the time will be extended for information only. Operating life is performed dynamically by applying 100KHz to the inputs of the devices under stress. Interim readouts are generally made after 168 hours. The junction temperature is kept below 175 degrees Celsius. Temperature cycling is performed from –65 to 150 degrees Celsius for 100 cycles per method 1010, condition C.

Temperature and humidity, usually referred to as 85/85 testing, is performed with devices biased. Both Vcc and ground are held at ground potential while the rest of the pins are biased to 5 volts. The name 85/85 comes from the temperature and humidity settings of 85 degrees Celsius and 85% relative humidity. Devices are stressed for 1000 hours.

Many other stress tests are performed including ESD testing. Monolithic Memories has a complete ESD control program in all post wafer fabrication areas. All necessary facilities are in place and training is carried out on a routine basis. Audits are also made of the manufacturing areas to ensure adequate control is maintained at all times.

The monitor program is intended to continually look at production products, packages and processes. Each month a series of product, package, and process combinations are selected to be monitored. Each quarter one product from each process and package is thus checked. Any defects are analyzed in the manner previously presented.

Whenever a major change is made to a product, process, or package, a re-qualification test is run. In addition to requalification, notification is made through the sales organization to those customers that have such notification requirements. Monolithic Memories generally notifies customers at least 90 days prior to implementation of a major change.

All the data are summarized and published twice per year in the Reliability Report. Copies of the report are available upon request.

## Electrical and Visual/Mechanical Quality

The basis of the quality improvement program has been thorough analysis of and corrective action for defective units found in four main areas. The Discrepant Material Reporting system stems from outgoing sampling performed by the QA Inspection group on each production lot of devices. QA uses a 0.065% AQL sampling plan per Mil Std 105D which generally results in a 200 piece sample. Any lot with a defect in the sample is returned to manufacturing for rescreening.

The Parts Per Million monitor is a unique effort to emulate the results that our customers find by using our products. The significant feature of the monitor is that the parts utilized are pulled from finished goods, the point closest to the customer but still within the factory. Devices are tested at room and hot temperature to arrive at an electrical PPM and visually inspected to arrive at a visual/mechanical PPM. In addition, samples are sent for programming and static burn in. In this way programming yield and infant mortality information are generated.

Defective devices from either of these factory locations are analyzed, summarized and have corrective actions generated by manufacturing and engineering. Formal reports are made to corporate management in the bimonthly Product Quality Review, which is a general meeting dedicated to reporting progress toward quality goals.

Two sources of customer feedback are the Customer Material Returns and failure analysis systems. Customer Material Returns (CMR) are generally returns by customers who are concerned with receiving credit or replacement of defective units they have found. These units are verified by a level 1 analysis as explained previously and summarized weekly and monthly in formal reports to management.

The fourth area of feedback is through the failure analysis system which has already been presented.

The results of the efforts have been the steady reduction in PPM levels to below 250ppm electrical, below 500ppm visual/mechanical, and below 0.1% infant mortality failures.

## Programmability

In addition to improving product quality, we have improved programming yields to higher levels. Improvement has come through algorithm revisions, random defect reduction, redesigns of products and programmer qualification. Once we began focussing on the programmer suppliers, we were able to better understand their programming processes and work much more closely with them to develop optimal programming algorithms. While this in itself helped to improve yields, we also made some design "tweaks" and manufacturing improvements that contributed further toward reaching levels generally above 99.5%. Post programming functionality was also improved.

Some of the issues that should be considered when considering performing your own programming are to maintain adequate calibration of programmers including replacing sockets after 10,000 insertions, inventory handling costs, human handling errors, and lack of post programming testing. Monolithic Memories can provide programmed product with minimum cost and handling that falls below 100ppm. This can be a significant reduction in the cost of quality and make it easier to move to a just-in-time manufacturing system.

## Internal Die Visual

It is Monolithic Memories' intent to supply die quality that meets or exceeds the Mil-Std 883, method 2010, class B. Toward this goal, Monolithic Memories has made significant improvements through random defect reduction in our manufacturing areas and through the use of statistical process control techniques. Significant defect reduction was gained by automating the assembly process to remove human handling. Similarly, automation in the wafer fabrication areas has also been effective. One significant improvement was made by putting pelicles on all photolithogra-

phic wafer masks. In addition to preserving the plate indefinitely, the pelicle causes any particle that should happen to fall onto the plate to be out of focus on the wafer. Therefore, a perfect print is made every time.

Die visual quality has improved to over 97% conformance to method 2010, class B in molded packages and to 100% conformance in hermetic packages. The die visual improvement has also contributed to the improvement in infant mortality and the improvement in programming and post programming functionality.

### Total Quality Process

Up to this point, the discussion has centered around detection and inspection to find quality problems and fix them. However, the long term trend is toward prevention. Statistical process control is a preventive measure that allows for building in quality by quickly heading off problems before they occur. Statistical process control (SPC) is the mainstay of a total quality approach.

Total Quality means the molding of attitudes through continual education, training and awareness in all areas, and to establish SPC in all manufacturing areas. The vehicles for accomplishing total quality are the bimonthly Product Quality Review, Quality Improvement Teams, Deming seminars, in house SPC training, use of SPC consultants, and an automatic data collection and analysis.

The actual use of SPC in manufacturing can be seen in both the U. S. and in the assembly facility in Penang, Malaysia. The assembly areas are leading the way with on line monitoring and real time charting in most operations. The wafer manufacturing areas have attained various levels of penetration.

The future holds more automation in store for SPC. By the end of 1987, we will have implemented a computer integrated system of data collection and analysis that will eventually allow for on line analysis, automatic data collection and automatic line control. We will be pushing ahead with more Taguchi techniques and with an SPC program to address non-production areas. And the bottom line will be to reduce our cost of quality through reduction in inspection and detection costs and to eliminate rework.

### Quality Partnership Program

Several years ago Monolithic Memories developed the Quality Partnership Program in order to facilitate quality improvement through enhanced feedback from a few customers that had good reporting mechanisms in place. Today, the program is basically the same with one very important exception. Monolithic Memories has attained quality levels below 250ppm as we measure ourselves. Through pareto analysis of your supplier base, you may not want to make the commitment necessary to sustain a true partnership relationship with a supplier that does not rank as one of your problems. But if you should, we are always ready to participate toward mutual improvement in quality. We have mechanisms in place to provide rapid and thorough failure analysis and outgoing inspection data on a regular basis. All that is needed is a commitment from our customers to provide us with rapid feedback and to be willing to work in tandem toward zero defects and minimum costs. The usual result of a successful partnership is reaching ship-to-stock and building a working relationship based upon trust and mutual understanding.

The quality programs described have been in existence for several years. The results have been dramatic. You will find us to be very honest and responsive to your needs. If you should need some information, please don't hesitate to contact us.

## Guidelines for Product Certification

### Characteristic Accountability

- Process Flow
- Product Specification
- Monitors and Controls
- Place of Manufacture
- Establish Correlation

### First Article Inspection

- Electrical Conformance
- Visual/Mechanical Conformance
- Variables Data

### Lot Monitoring Inspection

- Lot Acceptance or Defect Rate Maintenance
- Period of Time or Number of Lots Criteria

### Factory Audit

- Customer

### Certification

- Certification Maintenance Program
- Disqualification and Recertification Program
- Periodic Reports, Data, and Timely Feedback
- Special Outgoing Inspection and Verification
- Change Notification

## Product Assurance

### Quality Assurance-Military

- Facility Certification
- QPL Qualifications
- JEDEC 13/13.2 (Gov't Liason Committee)
- 38510 Quality Conformance Inspection

#### Product Acceptance

- Electrical Test Gates
- Visual/Mechanical Gates
- Traceability Verification
- Government/Customer Liason
- Final Outgoing Inspection
- Inspection Activity Reporting
- Verifies Conformance With MIL-M-38510 QCI Requirements

#### Quality Assurance Eng.

- Self Audits
- Corrective Action
- Calibration Control
- Customer Return Disposition, Reports and Failure Analysis
- Technical Resource
- Major Programs Support
- Division Quality Assessment Program
- Quality Partnership Program
- Configuration Control

#### Reliability Assurance Eng.

- Customer/Gov't Quality Conformance Inspection
- Package, Process and Reliability Studies (New/Revised)
- Division/Reliability Reporting
- CECC Reliability Programs

### Quality Control

- Incoming Inspection/Vendor Evaluations, Ratings and Corrective Action
- In-Line Procedural Audits (Fab Through Final Seal)
- In-Line Operational Audits (Fab Through Final Seal)
- In-Process Wafer Fab Inspection
- In-Process Wafer Sort Inspection
- In-Process Assembly Inspection
- Offshore Assembly Surveillance
- Offshore/Surveillance Correlation Program
- Analytical Lab Services (Di Water, Chemical Analysis)
- Environmental Audits (Temp., RH, Particle Counts and Flow Hood Velocity)
- In-Process CSI/GSI
- Quality Engineering Failure Analysis
- Statistical Control, Product Quality Analysis

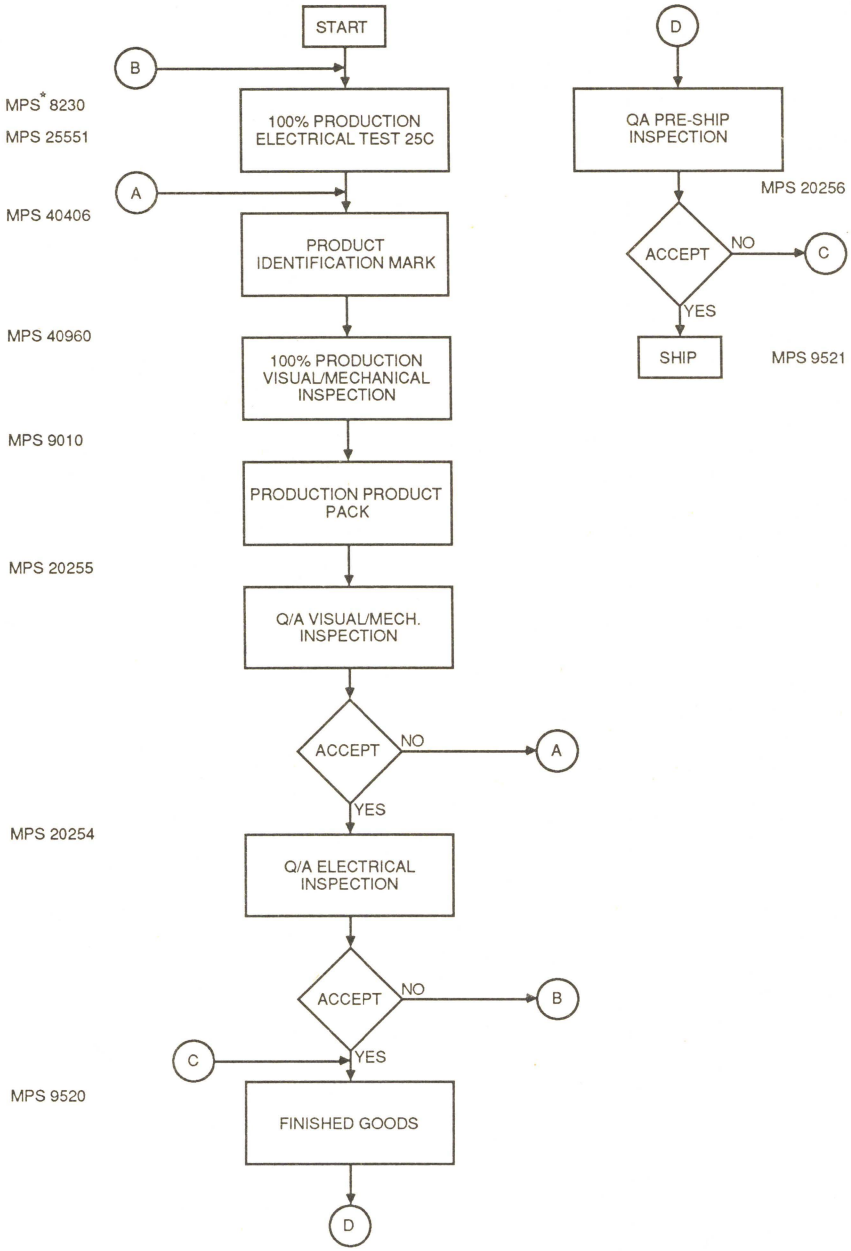
### Quality Assurance-Commercial

- Self Auditing
- Customer Material Returns Analysis and Corrective Action
- Discrepant Material Analysis and Corrective Action
- Technical Support
- Major Program Support
- PPM Monitor Support
- Quality Partnership Program
- Electrical Test Inspection
- Visual/Mechanical Inspection
- Final Outgoing Inspection
- Inspection Activity Reporting
- Commercial Traceability
- Conformance Verification of Special Commercial Specifications
- Document Control (Commercial)

### Reliability Assurance

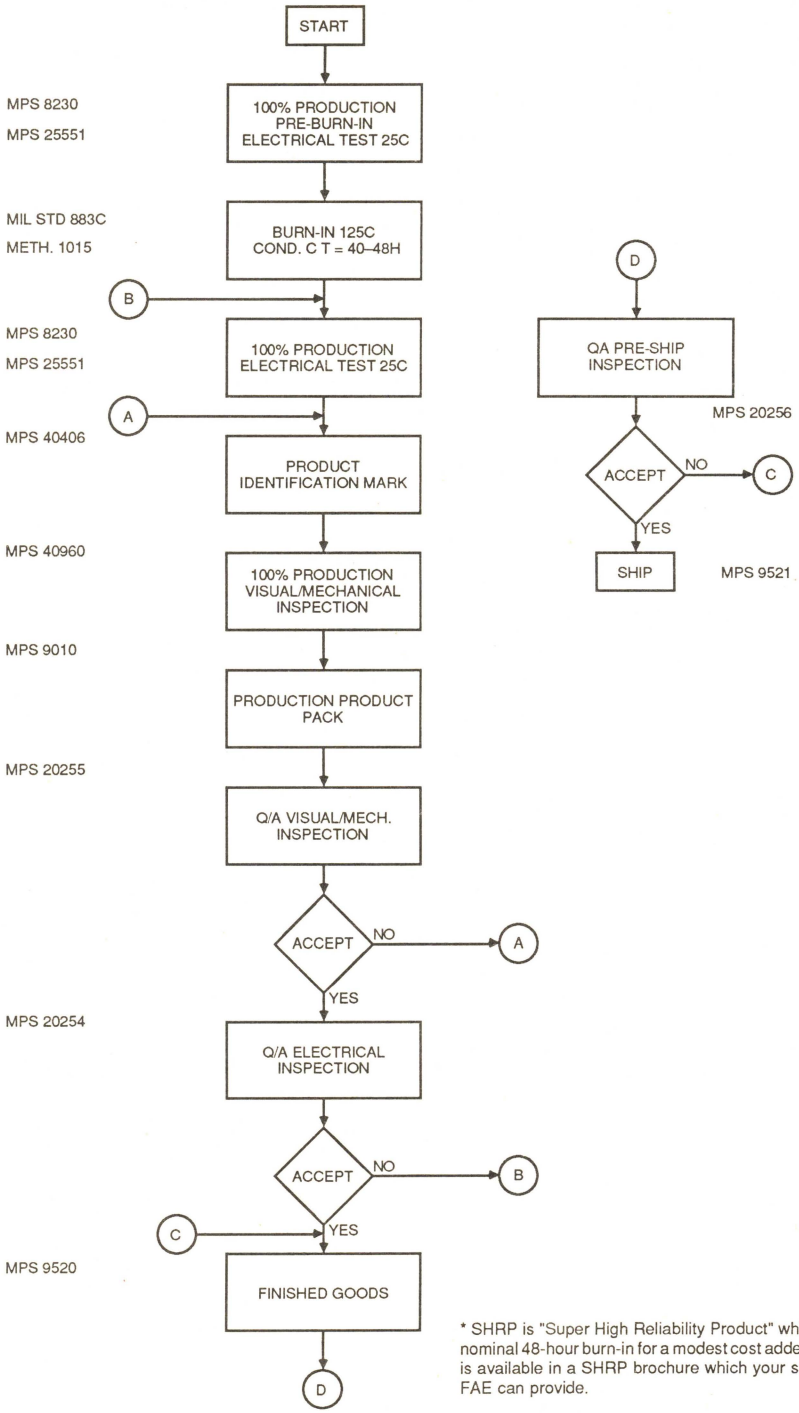
- Device/Design Qual (New/Revised)
- Package Qualification (New/Revised)
- Process Qualification (New/Revised)
- On-Going Device/Package Reliability Monitors
- Extended and Accelerated Life Testing
- Qualification Test Lab Per MIL-STD-883 Method 5005
- 38510 Device Qualification and 883 Quality Conformance
- Failure Analysis Lab (SEM/EDAX and Engineering Services)
- Early Failure Mode Detection and Corrective Action
- Reliability Reports (In-House and Field)

## STANDARD POST ASSEMBLY PROCESS FLOW




* Internal Manufacturing Process Specification Number

## SHRP* POST ASSEMBLY PROCESS FLOW




434 02

* SHRP is "Super High Reliability Product" which receives a nominal 48-hour burn-in for a modest cost adder. Information is available in a SHRP brochure which your salesperson or FAE can provide.

# Test and Finish Operations

Monolithic Memories performs the test and finish operations stateside in Santa Clara, Ca. and in Penang, Malaysia. Both facilities utilize state of the art electronic testing equipment as well as Electro-Static Discharge safeguards in all handling areas. Inventory is maintained and controlled via computerized database systems assuring one of the best on-time delivery systems in the industry. The product quality is monitored continually throughout the process to provide feedback necessary to support factory corrective actions.

## Electrical Testing/Pre Burn-In

The electrical test of integrated circuits starts long before a batch of parts is dispatched to the test area. Product and process characterizations must be performed to understand the parametric distributions and the test conditions that are necessary to provide full compliance to datasheet specification. When the customer order requires programming, functional test vectors are computer generated and evaluated for array coverage. Test software is validated for performance margin before being handed over to the Test area for use in production. The electrical test software is maintained through revision control and sign-off procedures.

Devices are delivered to the Test area for initial electrical testing. The product is 100% production tested to guarantee the databook requirements and functionality. The parameters specified include electrical and switching characteristics. In addition Monolithic Memories performs these tests at various ambient temperatures in order to eliminate marginal devices. Additionally, test program forcing conditions and test limits have been guardbanded to reduce the effects of system variability and parameter shift at temperature. All test equipment is calibrated to standards traceable to the National Bureau of Standards.

For reduced costs and improved quality due to the elimination of human handling errors, Monolithic Memories is incorporating bulk loading equipment into the Test and Finish areas. These machines can load and unload product from a handler with great speed, efficiency, and accuracy.

## Burn-In

Semiconductor failures over time are known to manifest themselves during the earliest stages of useful life. This phenomenon is known as 'Infant Mortality'. During burn-in, stresses are applied that accelerate failure for those devices which are prone to Infant Mortality. The elimination of these failures not only improves the reliability, but also results in substantial cost savings for system manufacturers by reducing rework and repair loading.

Typical conditions for burn-in are:

Static Condition C
Temperature: 125°C
Vcc: 5.25 V

The typical infant mortality phase is defined as 168 hours, with 75% of the defectives found in the first 48 hours.

Monolithic Memories has reduced mechanical handling defects during board load and unload by utilizing robotic handling/loading equipment.

## Electrical Test/Post Burn-In

The devices which are burned-in are again electrically tested to remove any failures that are a result of the Burn-in accelerated stresses. Production and engineering monitors have shown that the typical failure rate for this stress is approximately 0.05%.

## Marking

The devices are marked to provide identification of part number, MMI logo, assembly location, and date codes. In addition to standard marking, special customer required items are available. A photolithographic process is used to produce exceptional character clarity. Every lot is tested for marking permanency.

Traceability of all raw materials, equipment, operators and processes are identified by two unique date code numbers found on the top and bottom of the package. A six-digit code marked on the top-side of the device allows traceability of the test and finish operations, and an eight-digit bottom-side code for traceability of assembly processing and raw materials back to wafer lot.

## Visual/Mechanical Inspection

Visual and Mechanical inspection is performed on all production units. The inspection includes package outline dimensions, and lead and marking quality. With the aid of production monitors and statistical process controls, marginal processes are eliminated.

## Pack

Packaging of devices is no simple matter. One must provide mechanical strength, and identification of contents as well as electrical protection (from ESD). Monolithic Memories utilizes carbon impregnated boxes for the intermediate containers of a shipment, which act as faraday cages in dissipating the built-up electro-static charges. The box label has printed on it the part number, package type, quantity of devices, QA stamp of acceptance, specification number, date of pack, and bit pattern (if programmed). Many of the above items are also printed in machine-readable bar code format to assist in product movement and control.

## Quality Assurance Visual/ Mechanical and Electrical Inspections

The Quality Assurance department sample inspects the devices to a 0.065% AQL sample plan in accordance to Mil-HDBK-105D.

The visual/mechanical inspection consists of physical dimension checks along with lead, marking, and packaging verification to ensure quality.

Paperwork is also reviewed for accurate and complete processing to specification and customer requirement.

The devices are electrically tested to databook and/or specific customer conditions and limits to validate the electrical and functional integrity.

Any non-conformities that are found during QA inspection are verified and tracked through the appropriate product/assembly engineering group to obtain corrective action. This information along with inspection volume and failure rates is reported to management, operations and engineering regularly for long-term trend visibility and improvement.

# IMOX™ Product Technology and Reliability

In order to meet the next generation requirements for speed and density in PAL devices, an advanced bipolar technology has been developed called IMOX-III. Although IMOX-III represents a major breakthrough which will allow further scaling to the sub-micron region, the technology also shares many features in common with prior generations of technology, IMOX-II and IMOX-IIS.

The revolutionary breakthrough of IMOX-III is the use of reactive-ion-etched grooves, called slots, to isolate the transistors. These slots are 1.5 microns wide, over 6 microns deep, and are filled with dielectric material (Figure 1). Because the transistors are not isolated by junctions, space for depletion spreading is not necessary. Also, since the slots are etched anisotropically, thicker EPI layers can be isolated without increasing the isolation widths. Essentially, no density penalty is paid to achieve high breakdown voltages. Higher breakdown voltages are needed to support the programming voltages required to program fuses in bipolar PAL devices.

Smaller device sizes translate into faster circuits through smaller die sizes and reduced capacitances of active devices and metal interconnect. Another advantage of the slot isolation is reduced collector to substrate capacitance, which offers improved performance in many circuit configurations.

Overall, the IMOX-III process is a major step forward from IMOX-IIS. In addition to the slot isolation, stepper lithography and dry metal and via etching have been implemented, resulting in a dramatic reduction in device sizes. The slot isolation allows the silicon pitch to be reduced by one-third. The steppers and plasma metal etching allow the metal pitch to be shrunk by one-third also. Furthermore, the IMOX-III process was designed with a 20% shrink in mind. This scaling can be accomplished simply by shrinking the masks.

The IMOX-III process shares many familiar features with its predecessor, IMOX-IIS. Oxide-walled bases and emitters are used to reduce the size and parasitic capacitances of transistors. Ion implanted emitters and bases are used to achieve the profile control necessary for high performance transistors. The reliability of the transistor structure used in IMOX-III has been proven over millions of hours of high-temperature tests on products that use IMOX-II and IMOX-IIS processes.

Another key feature familiar to users of older generation IMOX PAL devices is the fuse technology. IMOX-III uses platinum silicide fuses, identical to the fuse technology used on older generation IMOX PAL devices. Programming yields are the highest possible, and programming times are extremely short (about 300 ns).

The IMOX-III technology also features two levels of metallization, as does IMOX-II and IMOX-IIS. However, with IMOX-III technology, both layers are stepper-defined and plasma-etched.

The IMOX-III technology is being applied to a family of high-performance PAL devices. The first of these is a "D-speed" 20-pin PAL IC, which runs at 10 ns. The table below shows the devices built on the IMOX processes.

| DEVICE | PROCESS | MAXIMUM PROPAGATION DELAY |
|---|---|---|
| AmPAL16R8 Family | IMOX-II | 25 ns |
| AmPAL16R8B Family | IMOX-IIS | 15 ns |
| AmPAL16R8D Family | IMOX-III | 10 ns |
| AmPAL18P8 | IMOX-IIS | 15 ns |
| AmPAL22V10 | IMOX-IIS | 25 ns |
| AmPAL22V10-15 | IMOX-III | 15 ns |
| AmPAL20XRP10 Family | IMOX-IIS | 15 ns |
| AmPAL23S8 | IMOX-IIS | 20 ns |

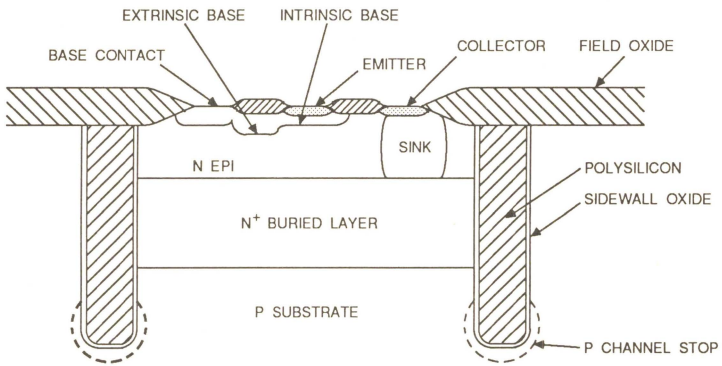



Figure 1. Slot Isolation

IMOX-III technology will enable third and fourth generations of PAL devices that will be significantly faster and more complex than the current devices. It will also reduce the cost of the new devices by significantly reducing die sizes or allowing more features to be added without increasing present die sizes. Faster and more complex PAL devices will permit system designers to build advanced computers, communications systems and instrumentation systems at a much lower cost.

## IMOX Product Reliability

IMOX bipolar Programmable Array Logic (PAL) devices are based on two key technologies with many years of high volume production experience behind them.

1. IMOX—The basic process technology employed is IMOX, an advanced ion-implanted, oxide-isolated structure. IMOX provides very high performance devices with predictable manufacturing yields. It has accumulated many millions of hours of life test history through its application to the Am27S series of PROMs and the Am2900 family of bipolar microprocessors.

   A comprehensive report on IMOX reliability titled IMOX RELIABILITY REPORT (AMD publication #03687A-MPR) is available for those interested in a detailed presentation on this subject.

2. Platinum-silicide fuses—This fuse structure was originally developed for use on junction-isolated PROMs. It quickly established a standard of excellence for high programming yields and long-term reliability. Several years ago it was applied to a new generation of ultra high performance PROMs based on the IMOX process.

This combination of IMOX and platinum-silicide fuses has an outstanding record of reliability which has been verified repeatedly through in-house life testing and by high-reliability customer qualification testing and system use.

IMOX PAL devices are fabricated with this same combined process technology. Not only is the technology for building PAL devices and PROMs the same, but also the programming algorithm and programming circuitry used to program the platinum-silicide fuses are the same in all characteristics of importance. The result is that the conditions seen by an IMOX PAL device fuse are the same as those seen by an IMOX PROM fuse.

Due to the common process technology, fuse design and fuse programming circuitry design, reliability and programming yield results are expected to be the same for PAL devices and PROMs. Data accumulated to date on PAL devices confirms this expectation.

This report describes the characteristics of the platinum-silicide fuse and programming conditions for the fuse, along with a description of the ongoing reliability monitor program.

## Platinum-Silicide Fuse

### Fusing Technique

IMOX PAL circuits are designed to use a programming algorithm which minimizes the requirements on the programmer yet allows the circuit to program the platinum silicide links quickly and reliably.

The sequence of events to program a fuse are:

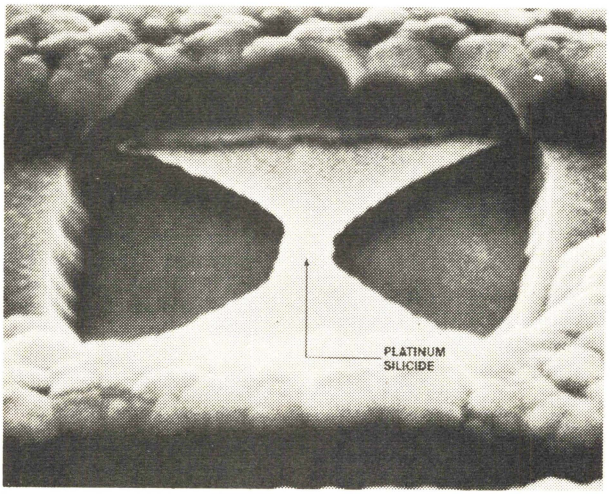
1. VCC power is applied to the chip.
2. The address of the fuse to be programmed is selected by TTL levels on the appropriate address pins.
3. The outputs are disabled. (Pin 1 serves this purpose on PAL devices).
4. The programming voltage is then applied to one output.
5. A fuse enable is accomplished by raising an input to a level above normal TTL operating voltage. (Pin 11 is used for this on PAL devices.) This action gates the current flow through the proper fuse, resulting in an open fuse in a few microseconds.
6. The output programming voltage is lowered and then removed.
7. The device is enabled and clocked if required. The output state then indicates whether successful programming has occurred. If programming has not occurred a sequence of much longer pulses is applied until programming occurs.
8. The sequence of 2 through 7 is repeated for each bit which must be programmed.

There are several advantages to this technique. First, the two high current power sources, VCC and the voltage applied to the output, do not have critical timing requirements. As the programming current is gated through the fuse actively, there is no dependence on the rise rate of the programming voltage. A fast application of programming current is desirable for optimum programming. Since the output programming voltage does not have to be applied rapidly, breakdown and latchback problems attributed to fast voltage rise times on the output are avoided.

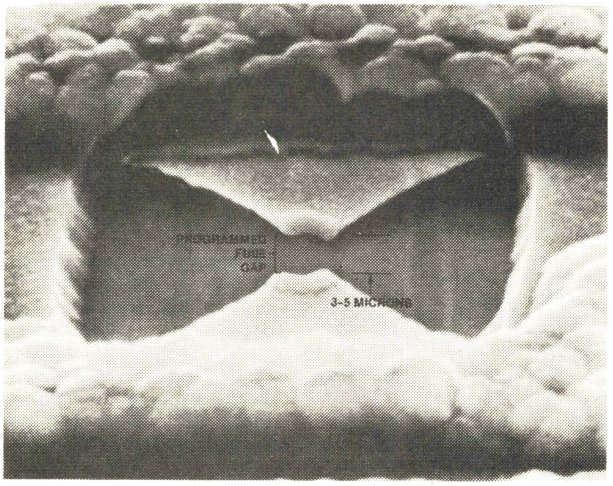
This programming procedure has a second major advantage. If the fuse does not open during the first programming pulse, longer programming pulses are used. With the platinum-silicide fuse, long programming pulses may be safely applied with no danger of developing a reliability problem. The algorithm can therefore be designed to minimize the time required to program by using a fast first pulse followed by a longer pulse if needed to program the occasional fuse that does not open with the first short pulse. Most devices do program satisfactorily with all short pulses.

## Fuse Characteristics

When a fast (less than 500 ns rise time) current pulse is applied to a fuse, the fuse voltage rises abruptly to a value determined by the room temperature resistance. However, it then quickly falls to a value of approximately 2 V. This value is nearly independent of the applied current. During the period of time the fuse is molten, the fuse current drops very abruptly to zero indicating the separation of the platinum-silicide into two distinct sections. Scanning Electron Microscope photographs of the resulting fuses (Figure 2) indicate that the typical case is a sharp clean separation in excess of a micron. This separation occurs in the center of the fuse because the "bow-tie" structure (Figure 3) concentrates the energy density in the center away from the aluminum interconnect lines. The energy density in the center of the fuse creates temperatures substantially greater than those required to melt the silicide. Melted material is then "wicked" from the center of the fuse to either side due to surface tension.




Unprogrammed Fuse




Programmed Fuse

Figure 2. Scanning Electron Microscope Photo–Unprogrammed and Programmed Fuses

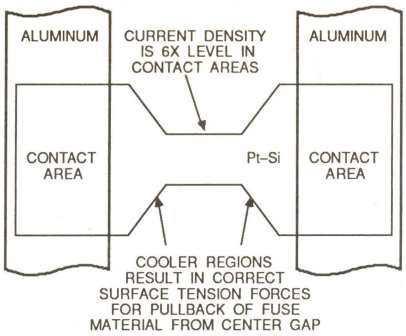



**Figure 3. Bow–Tie Fuse Design**

## Reliability Testing Data

Data on the reliability of PAL and PROM devices with platinum-silicide fuses is gathered via the Reliability Monitor Program (RMP). The RMP is an ongoing program conducted on all device types across all product lines, and is designed to ensure that all IMOX devices meet acceptable reliability levels. A summary of the RMP tests for hermetic and plastic molded packages are shown in Tables 1 and 2.

Data on IMOX PAL and PROM devices has been gathered over millions of device hours and more than 40 billion fuse hours of high temperature operating life tests (HTOL). The life test circuits used in this work conform to MIL-STD-883 method 1005 conditions C and D. This data indicates a projected unit failure rate (at 60% confidence) of 0.0002%/1000 hrs. at 70°C.

Results of the IMOX RMP are updated periodically and can be obtained through inquiry to any of the Sales Offices listed in the back of this handbook.

| TEST | CONDITIONS | TYPICAL SAMPLE SIZE |
|---|---|---|
| Infant Mortality | 160 hours at 125°C ambient. Initial and end-point electrical tests. | 300 |
| Operating Life | 1000 hrs (1160 total) at 125°C ambient. Initial and end-point electrical tests. | 120 |
| Temperature Cycle | 1000 cycles, (–65°C to 150°C), 30 min/cycle. End-point-hermeticity and electrical tests. | 50* |
| 150°C Operating Cycle | 1000 hours at 150°C ambient. Initial and end-point electrical tests. | 50 |

* These units are hermetically tested prior to commencement of test.

**Table 1. Reliability Monitor Program for Devices in Hermetic Packages**

| TEST | CONDITIONS | TYPICAL SAMPLE SIZE |
|---|---|---|
| Infant Mortality | 160 hours at 125°C or 85°C ambient (Tj <150°C nominal). Initial and end-point electrical tests. | 300 |
| Operating Life | 1000 hrs (1160 total) @ 125°C or 85°C ambient (Tj <150°C, nominal). Initial & end-point electrical tests. | 120 |
| Temperature And Humidity | 85°C/85% RH/low power bias, 500 hours and 1000 hrs. Initial, interim, and end-point electrical tests. | 50 |
| Temperature Cycle | 1000 cycles: –65°C to 150°C, 30 minutes/ cycle. High temperature (75°C min) functional end-point electrical test. | 50 |
| Pressure Cooker | 121°C, 15 psi, 160 hours, unbiased, initial end-point electrical test. | 50 |

**Table 2. Reliability Monitor Program for Devices in Molded Packages**

## IMOX Product Testability

Thorough testing of programmable logic devices by the manufacturer is important to both the performance of programmable logic and its cost of use.

Field programmable logic devices are different from other semiconductor products in that the user must complete the manufacturing process by programming and functionally testing the parts.

Programming is normally accomplished on commercially available programming equipment. Functional testing may be performed on a programmer, on automatic test equipment or at the board or system level. Figure 4 illustrates where device failure detection can occur. Clearly, the cost implications of failure become more serious with each advancing step.

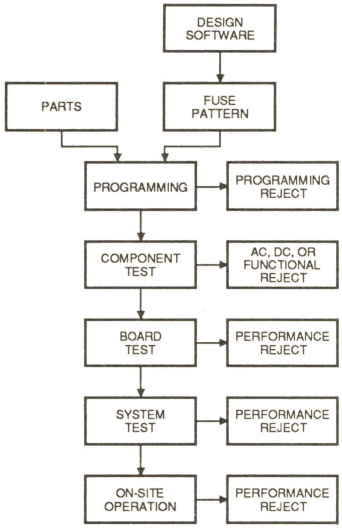



**Figure 4.**

As a result of assuming the responsibility of programming and test, the user gains all the benefits of a custom function with the cost and availability advantages of a standard product. However, the user must also deal with those parts that do not program successfully or do not function to advertised specifications after programming.

Testing before shipping can make a difference to the user in:

1. Programming yield
2. Post-programming functional yield (PPFY)
3. Uniformity of performance

This paper describes the techniques used on IMOX process PAL devices to allow testing of these three important attributes on every device before shipment to the user.

## Programming Yield

Programming yield is the measure of the success of the programming operation. Large volume users of programmable logic keep records of the programming yield history of their suppliers' parts. Programming yield is considered by these users to be an important element in judging the overall suitability of different suppliers' parts.

## Post-Programming Functional Yield

Experienced PROM and EPROM users are sometimes puzzled by the fact that not all programmable logic devices function correctly even though they have successfully completed a programming operation and fuse verification check.

With PROMs, a one-for-one relationship exists between address states and programming elements (which can be fuses, floating gate MOS devices, open-base NPN transistors, etc.) That is, the state of each output for each address is dependent on the condition of only one fuse. Sensing a desired fuse state after programming therefore practically guarantees correct functional operation (at least at the voltage and temperature conditions of the programming operation).

With programmable logic devices the relationship between programming success and post-programming functionality is not one-for-one. Except for the simplest of patterns and devices, the relationship is highly complex. Feedback buffers allow the creation of more than one level of logic; latches, counters, shift registers, and even oscillators can be created. Special fuse functions such as polarity control, output enables, register/combinatorial path selection and buried registers complicate the relationship further.

This is the power of programmable logic—but the testing challenge that results from this versatility can be substantial. Logic states for programmable logic devices can depend on multiple fuses. The fuse verification procedure that examines each fuse uniquely is therefore not sufficient, as it is with PROMs, for guaranteeing functionality.

All programmable logic devices contain special on-chip programming circuitry and modes to allow programming and verification of each individual fuse. The complexity of programming may vary significantly, but all have one thing in common—successful programming by itself cannot guarantee functionality.

The user's job does not end then with the programming operation. To be assured of a functional part, a comprehensive set of test vectors must be applied to the part. Many device programmers accept test vectors along with fuse programming vectors and will apply the test vectors to the part following the programming operation. The PRELOAD feature greatly simplifies the test generation problem for registered parts.

## Uniformity of Performance

The buyer of a programmable logic part has the right to expect that the performance specifications appearing on the manufacturer's data sheet will be met for all legitimate applications of the part. This applies to each and every logic path and function.

A glance at the logic diagram for an unprogrammed part shows that, with the array in its unprogrammed state, no amount of activity of the inputs can make any output switch. Without any programmed fuses, the AND gates see both the true and complement of all inputs.

If post-programming performance is to be guaranteed with absolute confidence, test circuitry must be provided to allow each path to be tested to data sheet performance.

## Approach to Designing in Testability in IMOX PAL Devices

The approach to the the design of IMOX programmable logic was strongly influenced by the goal to provide users with the best programming yield, post-programming functional yield, and uniformity of performance.

Designing programmable logic can be viewed as a three-dimensional task involving high-performance logic design, fuse programming circuit design and test circuit design.

The first dimension is the design of a high-performance logic circuit with SSI/MSI competitive switching speeds and very high output drive for bus environments.

The second dimension of programmable logic design is the programming circuit design. The emphasis of this design is to provide circuitry that will deliver large programming currents to individual fuses. Special decoders, demultiplexers, buffers and mode select circuitry are needed. The circuits need not be fast since programming occurs at microsecond speeds. Because the circuitry is not used after programming, it is desirable that it consume power only during programming and not during operation. Since large voltages are required to generate programming current, survival under high voltage is also required. All of these requirements are quite different from the logic circuit requirements but must be achieved within the same part.

Testability is the third dimension of programmable logic design. This overlay of circuitry provides the means to exercise the part through all of the possible paths that might be activated by programming. Test circuitry is also needed to insure that the programming circuitry will function properly. Testability is thus important to achieving high programming yields, post-programming functionality, and performance to data sheet specifications through all possible paths.

The unique challenge of programmable logic design is to integrate these three dimensions in the most efficient manner.

## Testability in the Programming Circuitry

Good programming yields are in the high ninety percent range. IMOX PAL device programming yields are typically higher than 98%.

Three things contribute to the high success rate in programming IMOX fuses:

1. Uniform fuse cross sections.
2. Pretesting of the current delivery and sink capability of column drivers and row drivers through use of wafer sort test pads.
3. Sample fusing of test rows.

### Uniformity of Fuse Cross Sections

The IMOX process gives consistently uniform platinum-silicide fuse cross sections. Uniformity is monitored by measuring fuse resistance test patterns on a sample basis in every wafer lot. The data is processed for mean and standard deviation and trend plots are maintained. Material not meeting fuse width control limits is scrapped.

### Testing for Fusing Current Delivery Capability

On every IMOX PAL device there are two extra pads that are probed at wafer sort. These extra pads are used to gain access to the fuse array for special testing at wafer sort. The connection of these pads to the fuse array is shown in Figure 5.

The programming process involves selection of individual column and row drivers to deliver and sink programming current through selected fuses. The extra test pads allow easy access for individually testing the source and sink capability of each column and row driver. Also a reverse leakage check of all of the Schottky diodes in the array is possible by applying bias between the pads. Without the test pads, all of these tests would be impossible or would have to be accomplished in a less direct and less effective manner.

### Sample Programming

To further assure programmability, the IMOX PAL devices include an extra test input buffer with fuses connected to each of the array columns.

Programming one test buffer fuse per column accomplishes two important things. First, a sample fuse has been programmed using each of the column drivers. The sample fuse is exactly the same dimension as all of the normal array fuses, and the test buffer drivers sinking the programming current are identical to all of the normal drivers. Before shipment each IMOX PAL device has had a sample of fuses programmed on the test buffer. For example, 64 fuses are programmed on the test word of every AmPAL16L8, one per product term.

The second purpose in programming the sample fuses is to create a pattern for AC and functional testing.

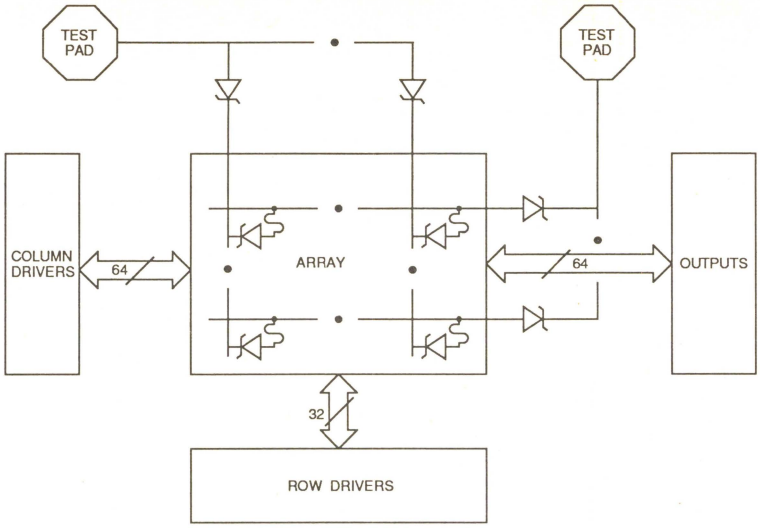



Figure 5.

## Testability to Guarantee Functionality After Programming

A typical PAL device, the AmPAL16R4, is shown in Figure 6. Not shown in the logic diagram are the components located at each horizontal and vertical line intersection. For IMOX PAL devices, a fuse and a Schottky diode reside at each cross point as shown in Figure 5.

The horizontal or "Product Term" line is then the common anode connection for a 32-wide diode AND gate. The user's job is to figure out which of the 32 inputs should be connected to the AND gates. The inputs not needed must be disconnected by programming the fuse shown in series with the diode.

The obvious problem from a manufacturer's test standpoint is: How can it be guaranteed through testing that the device will work after fuses are programmed? If the only logic in the device were that shown in Figure 7, testing would be nearly impossible. With 16 LOW levels and 16 HIGH levels presented to each AND gate, the LOWs win. All 64 AND outputs are thus always stuck LOW, and there is no way to get the output to toggle for AC or DC test purposes. This is the raw state of any device before programming.

### Necessary Testability Requirements

Something more is needed in every PAL device to assure close to 100% functional yield after programming. The IMOX PAL devices have an overlay of test circuitry that accomplishes the following:

1. Each input and feedback buffer can be checked for functionality.
2. Each of the AND gates can be switched HIGH and LOW and uniquely sensed by an output.

These two tests are necessary to the guarantee of close to 100% post-programming functional yield.

Under normal operating conditions the test circuitry is inactive and consumes very little power. Supervoltages cause it to come alive. Supervoltages are levels substantially higher than Vcc so that under normal operating conditions accidental activation of a test mode cannot occur.

In this paper a double line on the input side of a logic symbol indicates that the HIGH level must be a supervoltage to activate it.

### Checking the Input and Feedback Buffers

Functionality of the input and feedback buffers is checked with the aid of the extra AND gate dedicated to this function. Figure 7 illustrates the AND gate and its associated enabling circuitry.

The non-inverting or true side of each input and feedback buffer is connected to the special test AND gate. The AND gate is activated by a supervoltage on one of the input pins. The function actually takes two activating inputs to implement since the use of one for activation prevents that pin from being tested for functionality. Having an alternate pin to activate the function solves this problem.

Only the non-inverting side of each buffer is hooked up to the AND gate because each buffer is constructed from two inverters in series. The first inverter must work for the second one to work, so that checking the second one is sufficient to prove that they both work.

The feedback from the output used for the test cannot be fed to the test AND gate; such a connection would make the test output oscillate. For this reason its feedback input is not connected and is tested by creating another test AND gate on a different output and routing it there.

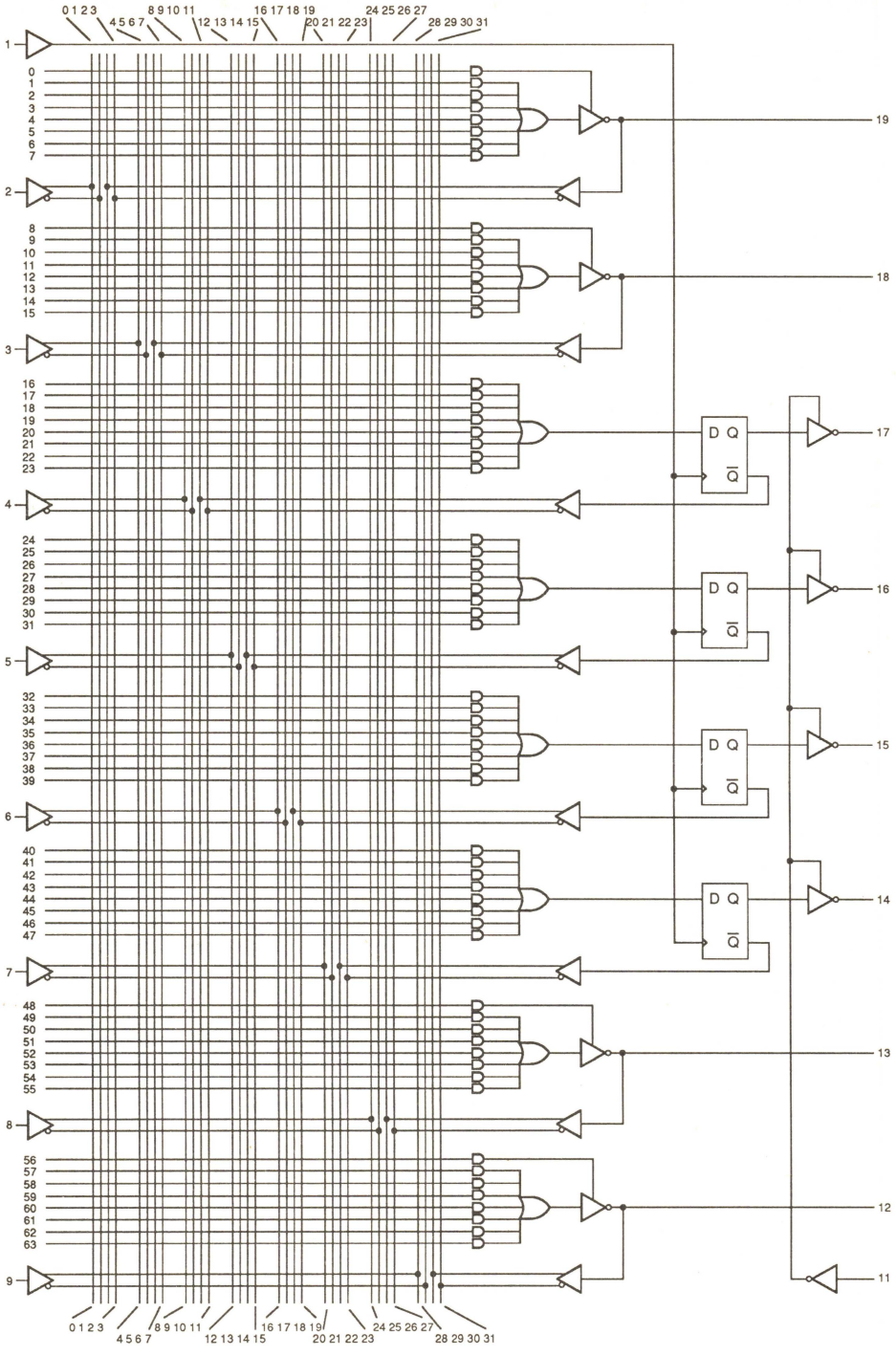
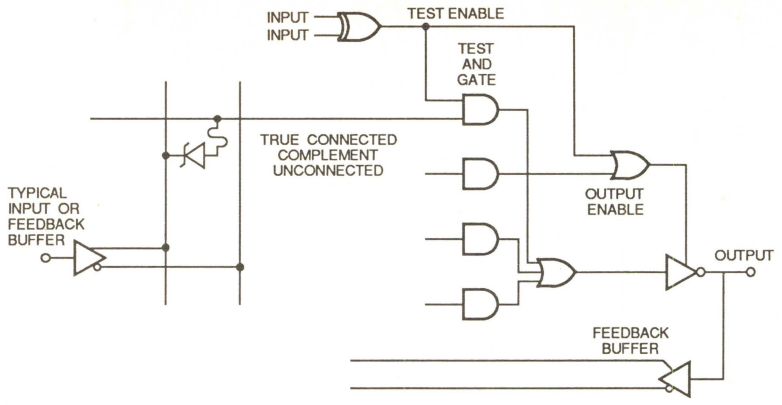



Figure 6.

469 03




469 04

**Figure 7.**

Since the special AND gate used to test all of the buffers is identical to those used in the normal operating path, switching each input through this path provides the means for testing the switching performance of each buffer.

### Testing the AND Gates

The next important test requirement is to make sure that all of the AND gates work and will switch at data sheet speeds. This test challenge is little more complex.

What is needed in this case is:

1. A means of decoding one AND gate at a time in each output.
2. A way to force all input and feedback buffers to a HIGH level on both true and complement outputs.
3. A special input of identical design to a normal input that can be used to switch the decoded AND gates.

These requirements are met by the circuitry shown in Figure 8.

The decoder to select one AND gate at a time in each output serves a dual purpose. It is the same decoder that provides unique selection of product term lines for programming and fuse verification. It responds to binary combinations of TTL signals at three input pins; only one of the eight outputs will go high at a time, thereby isolating each AND gate.

The special test input that is used in this mode also serves a dual purpose. It was mentioned earlier in this paper that a programming sample was performed on each part. This special test input is the input that carries the test fuses. During the sample programming operation the fuses are programmed in a pattern that allows switching of all 64 AND gates, one in each output, for each of the eight decode states.

The input to the special buffer for AND gate testing is one of the normal input pins, but the buffer is inactive for normal operation and must be activated by supervoltage levels applied to two other inputs.

The supervoltage levels also provide the signal to force all of the buffer outputs HIGH, which is one of the three necessary requirements for AND gate testing.

Since the design of the special buffer is identical to all of the normal input buffers, it serves as a surrogate buffer for speed-testing all of the AND gates. In the AND gate test mode, all eight outputs are switched at once, since one AND gate is selected in each output. For registered outputs the AND gate switching path provides a means of testing setup and hold times.

## Summary

All IMOX programmable logic devices have designed-in testability and are achieving yields of greater than 98% for programming and better than 99.9% functional and AC test yields after programming. Even higher goals have been set for future products.

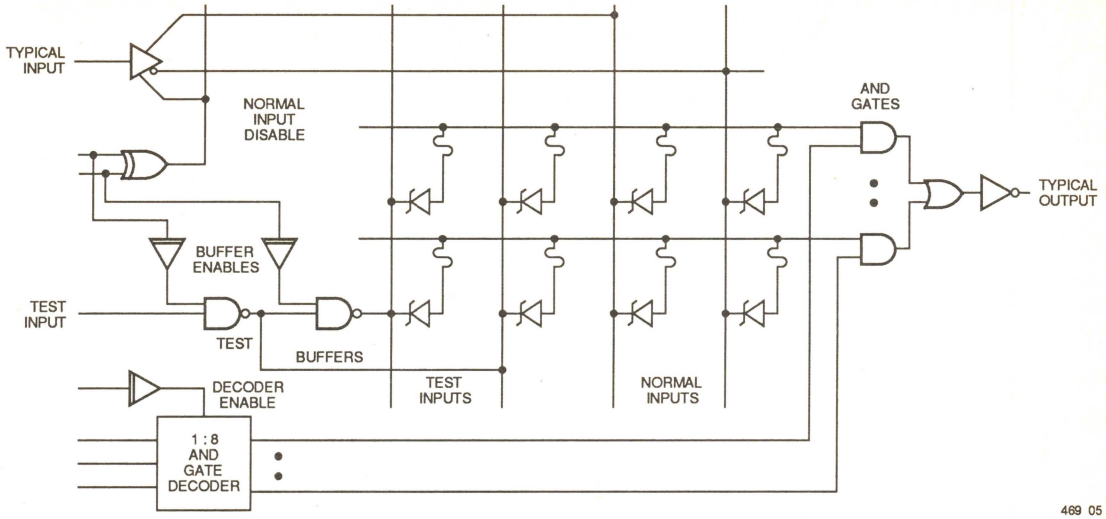

TYPICAL INPUT
NORMAL INPUT DISABLE
AND GATES
TYPICAL OUTPUT
BUFFER ENABLES
TEST INPUT
TEST
BUFFERS
DECODER ENABLE
TEST INPUTS
NORMAL INPUTS
1 : 8 AND GATE DECODER
469 05

Figure 8.

# ECL Technology

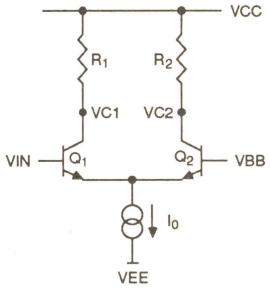
Emitter-Coupled Logic (ECL) delivers high speed, high input impedance, and low output impedance. These features are ideal for system designers who want to improve system performance. ECL achieves high speed by operating the transistor in a non-saturation mode, so that the storage-time delay associated with saturating logic is not present. When high input impedance is combined with low output impedance, large fan-in and fan-out can be achieved; in addition, low-impedance transmission lines can be driven.

## Current Switch

The differential amplifier (Figure 1) is the basic building block for ECL logic; it functions as a current switch. Current $I_0$ is steered either through resistor $R_1$ or $R_2$ depending on the input voltage (VIN). A difference of 150 mV between VIN and VBB will cause the current $I_0$ to flow entirely through the transistor with the higher base-emitter voltage (VBE), due to the exponential relationship between the collector current and VBE.

When VIN is 150 mV less than the reference voltage (VBB), the collector voltage of $Q_2$ (VC2) will equal VCC-$\alpha I_0 R$ when $R_1=R_2=R$, and the collector voltage of $Q_2$ (VC1) will equal VCC. When VIN equals VBB, the two collector currents will be equal and VC1=VC2=VCC-$\alpha I_0 R/2$. When VIN is 150 mV higher than VBB, VC1=VCC-$\alpha I_0 R$, and VC2 = VCC. Although the switching threshold is 300 mV centered about VBB, the signal swing is made larger (approximately 850 mV) to provide noise immunity and to provide for differences between input thresholds of one circuit and output voltage levels of another. The values of R and the current source are chosen to determine the voltage swing and ensure the charging and discharging of parasitic capacitances at a given switching rate.

If we consider VCC to be a high logic level, and VCC-$\alpha I_0 R$ to be a low logic level, then VC1 will always be the inverse of VC2. That is, they are complementary outputs.

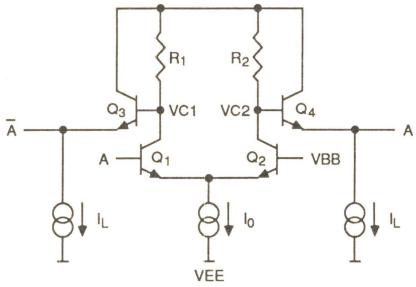



436 01

**Figure 1. The Basic Differential Amplifier**

## Emitter-Follower

To keep the current switch out of saturation, VIN must not be greater than VCC-$\alpha I_0 R$. For matched switching speed the voltage swing should be centered around VBB, which must also be below VCC-$\alpha I_0 R$ by at least one half the voltage swing. To meet these restrictions, an emitter-follower is added to the current switch (Figure 2). This shifts the level of VC1 and VC2 down by a diode, and allows the logic gate to have compatible input and output levels. The emitter-follower also isolates the collector switch nodes from the output load, and provides the low output impedance that is beneficial to this logic family. On outputs, the emitter-follower can drive a 50 Ω load to –2.0 V. This allows a terminated transmission line, preventing reflections.

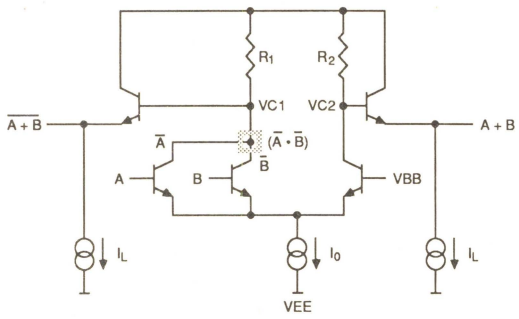



436 02

**Figure 2. Gate with Emitter-Followers**

## OR Logic

The buffered switch in Figure 2 functions as a buffer or an inverter. By adding a second input transistor to the buffered switch (Figure 3a), a wired-AND of the collector impedances is formed at VC1. By using DeMorgan's rule, this AND function is inverted into the NOR of the two inputs. The inverse of VC1, the OR function, is generated at VC2.




436 03a

**Figure 3a. OR/NOR Gate**

Additional parallel input transistors can be added to this gate to create multiple input gates (Figures 3b and 3c). The limit of the number of parallel input transistors is set by the speed of the gate; additional input transistors add more capacitance to the collector switch node forming the NOR logic. Given enough capacitance difference between the two collector switch nodes, a skew in the two outputs will occur.

## Current Source

The current source used in 10KH and 100K ECL circuits is illustrated in Figure 4. The source current $I_o$ is set by the reference voltage VCS, the emitter resistor $R_3$, and the base-emitter voltage of $Q_3$. VCS is internally generated and is at a fixed voltage with respect to the negative supply VEE. The source current is

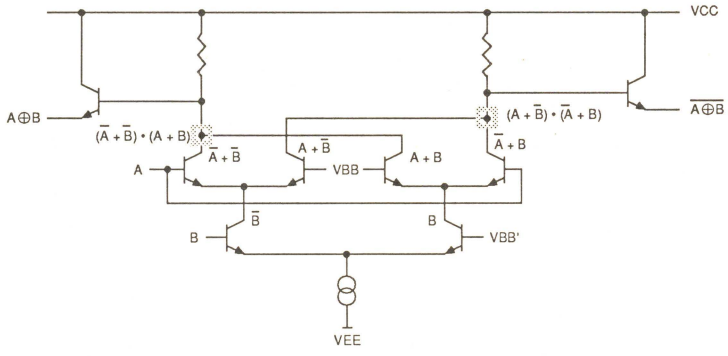



Figure 3b. 2-Input XOR/XNOR Gate

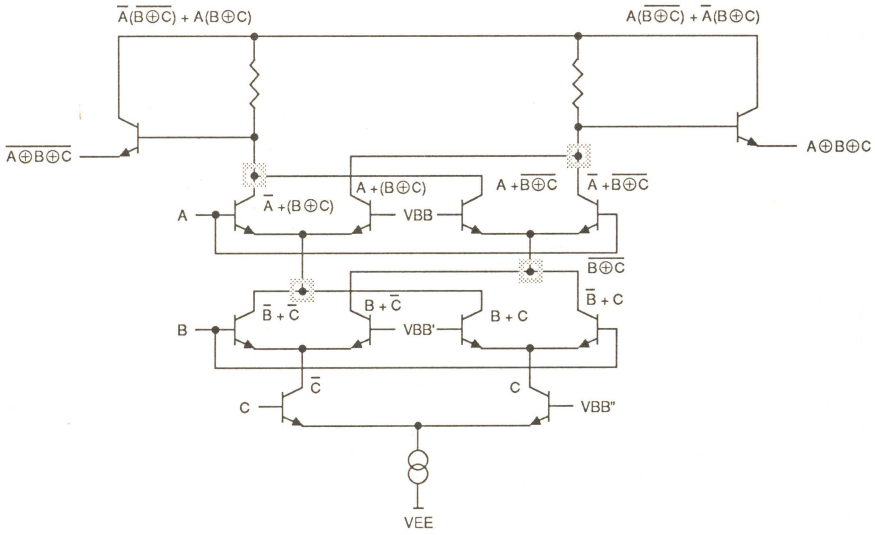



Figure 3c. 3-Input XOR/XNOR Gate

independent of the VEE supply voltage because of this fixed voltage. The output levels are primarily determined by the collector voltages of $Q_1$ and $Q_2$. As discussed earlier, these voltages are VCC-$\alpha I_0 R$ or VCC. This relationship between the output levels and the source current makes the output levels practically insensitive to VEE variations. Thus these ECL circuits are said to be voltage compensated. The variation of output voltages with respect to VEE for the 10KH and 100K families is shown in Table 1.

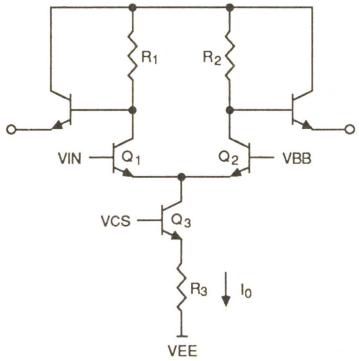



Figure 4.

| | 10KH | 100K |
|---|---|---|
| ΔVOH / ΔVEE mV/V | –20 | 7 |
| ΔVOL / ΔVEE mV/V | 20 | 15 |
| ΔVBB / ΔVEE mV/V | 10 | 10 |

Table 1. Voltage Sensitivity

## Input Threshold Regulation

The input threshold region is centered around VBB. VBB is internally generated and is at a fixed voltage with respect to the positive supply VCC. Variations in VEE have minimal effect in the value of VBB. The relationship between VBB and VEE is also shown in Table 1.

## 10KH Temperature Tracking

The output levels of 10KH circuits vary over temperature. The input threshold voltage VBB also varies over temperature to track the output variation. The temperature tracking characteristics of the 10KH output levels and input thresholds are shown in Table 2.

| | 10KH | 100K |
|---|---|---|
| ΔVOH / ΔT mV/°C | 1.3 | <0.1 |
| ΔVOL / ΔT mV/°C | 0.5 | <0.1 |
| ΔVBB / ΔT mV/°C | 1.0 | <0.1 |

Table 2. Temperature Sensitivity

## 100K Temperature Compensation

The output levels and input thresholds of 100K circuits are temperature compensated. The input threshold is compensated in the bias network by referencing it to the extrapolated energy band-gap voltage of silicon (VGO = 1.3 V), which is generated in an on-chip regulator. The output levels are compensated by a cross-connect network in the current switch and a temperature-regulated current-source driver (Figure 5). The cross-connect network adds a negative temperature coefficient to the base of $Q_4$ when the true output of the gate is in the VOH state, which compensates for the positive temperature coefficient developed by the base-emitter junction of $Q_4$. Additionally this same circuit adds a positive temperature coefficient to the base of $Q_4$ when the gate is in the VOL state, thereby compensating for the dominant negative coefficient introduced by the gate's current source. The temperature dependence of the 100K output levels and input thresholds is shown in Table 2.

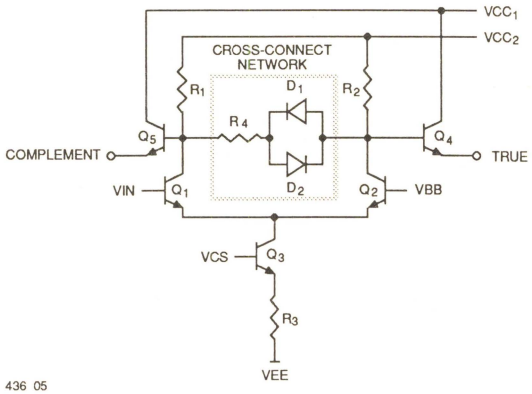



Figure 5. Output Temperature Compensation

## Voltage Supply Range

Because the outputs switch high currents very rapidly, they can generate a lot of noise on the VCC line. The circuitry operates with small voltage swings, so this noise can disrupt the rest of the circuit. For this reason, multiple VCC lines are used to isolate the internal "clean" VCC from the VCC that drives the outputs, (Figure 6). If the device has many outputs, several "dirty" VCC pins may be used. As a general rule, provide one VCC for each group of four outputs.

Because of the necessity for a clean VCC supply, it is desirable to connect it to the most stable voltage in a system, which is normally ground. Thus VEE is normally negative. The normal ECL VEE supply ranges are shown in Table 3.

| | NOMINAL (V) | RANGE |
|---|---|---|
| 10KH | –5.2 | ±0.5% |
| 100K | –4.5 | ±0.3 V |

Table 3.

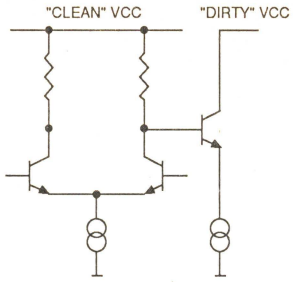
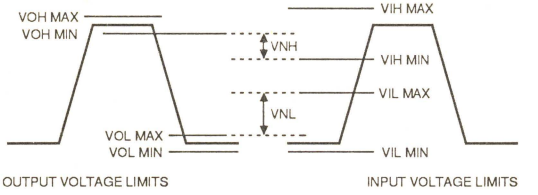



Figure 6.

436 07




436 06

Figure 7. Noise Margins

## Noise Margins

Noise margins between circuits with different supply voltages do not degrade more than 30 mV because of the insensitivity of both the output voltage and the threshold voltage to changes in VEE. This simplifies the requirements for the system power regulation and distribution. The minimum noise margin is defined by the difference between the Min VOH and Min VIH, or VNH, and the difference between the Max VOL and Max VIL, or VNL (Figure 7).

In Table 4 the input and output electrical characteristics are shown for the 10KH family. Both VNH and VNL are 150 mV for two parts at the same temperature. With one part at 0°C driving another part at 75°C the VNL drops to 50 mV. This happens because the input thresholds and output voltages of 10KH circuits are not temperature compensated. The 100K circuits have temperature compensation and provide better noise margin for parts operating at different temperatures. Table 5 shows this; notice that input and output characteristics are specified for difference in power supply and not in temperature as is Table 4 for 10KH characteristics. The worst-case VNH for 100K parts is 115 mV. This occurs when one part has a VEE of –4.8 V and is driving a part whose VEE is at –4.2 V. The worst-case VNL for 100K parts is also 115mV. This occurs when one part has a VEE of –4.2 V and is driving a part whose VEE is at –4.8 V. The system designer must also take into account any variations in VCC from one part to the other as this will have direct affect on the system performance with respect to noise.

## 10KH Electrical Characteristics $V_{EE}$ = –5.2 V ± 5%, Outputs Terminated With 50 Ω to –2.0 V

| SYMBOL | PARAMETER | 0° | | 25° | | 75° | | UNIT |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $V_{OH}$ | High output voltage | –1.02 | –0.84 | –0.98 | –0.81 | –0.92 | –0.735 | V dc |
| $V_{OL}$ | Low output voltage | –1.95 | –1.63 | –1.95 | –1.63 | –1.95 | –1.60 | V dc |
| $V_{IH}$ | High input voltage | –1.17 | –0.84 | –1.13 | –0.81 | –1.07 | –0.735 | V dc |
| $V_{IL}$ | Low input voltage | –1.95 | –1.48 | –1.95 | –1.48 | –1.95 | –1.45 | V dc |

Table 4. 10KH Input and Output Characteristics

## 100K Electrical Characteristics $0°C < T_{CASE} < 85°C$, Outputs Terminated With 50 Ω to –2.0 V

| SYMBOL | PARAMETER | TEST CONDITIONS | | | | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | $V_{EE}$ (V) | MIN | TYP | MAX | |
| $V_{OH}$ | High output voltage | | –4.2 | –1.020 | — | –0.870 | mV |
| | | | –4.5 | –1.025 | –0.995 | –0.880 | |
| | | | –4.8 | –1.035 | — | –0.880 | |
| $V_{OL}$ | Low output voltage | | –4.2 | –1.810 | — | –1.605 | mV |
| | | | –4.5 | –1.810 | –1.705 | –1.620 | |
| | | | –4.8 | –1.930 | — | –1.620 | |
| $V_{IH}$ | High input voltage | | –4.2 | –1.150 | — | –0.880 | Vdc |
| | | | –4.5 | –1.165 | — | –0.880 | |
| | | | –4.8 | –1.150 | — | –0.880 | |
| $V_{IL}$ | Low input voltage | | –4.2 | –1.810 | — | –1.490 | Vdc |
| | | | –4.5 | –1.810 | — | –1.475 | |
| | | | –4.8 | –1.810 | — | –1.490 | |

Table 5. 100K Input and Output Characteristics

# CMOS HiPAC Technology

## Introduction

CMOS has recently become one of the most important technologies for VLSI circuits. Monolothic Memories has successfully developed different versions of advanced 1.2-micron CMOS technology with the capability to manufacture High-performance Programmable Array CMOS (HiPAC) products. The HiPAC technology advocates many advantages of CMOS circuits, including very low power consumption, high noise margin, excellent speed performance, high device density, and improved reliability. In addition, the HiPAC technology accommodates a double-poly structure to fabricate the conventional EPROM cell, which provides the programming capability for programmable CMOS products. The EPROM cell is electrically programmable, then erasable by ultraviolet light, and reprogrammable. Such reprogrammable characteristics allow a comprehensive test of the parts to guarantee 100% programming yield. Any defective parts can also be screened out during the manufacturing process. This assures the excellent reliability of programmable CMOS products. The programmed data are stored in the double-poly structure, and are not visible even if the packaged part is opened. Security of the programmed data is thus enhanced. CMOS products are also well protected from potential destructive damage caused by electrostatic discharge and latchup phenomena.

## CMOS HiPAC Technology

The advanced HiPAC technology developed is an N-well double-layer poly, single-layer metal process based on a p-type substrate. A self-aligned lightly doped drain-source (LDD) has been incorporated for the regular n- and p-channel transistors. The LDD structure reduces the gate to drain-source overlap capacitance for speed improvement. It also increases the field induced drain breakdown voltage, increases the punchthrough voltage, decreases the impact ionization, and thus minimizes the hot electron injection. All these aid to improve the device reliability significantly. Figure 1 gives a process cross section of the HiPAC process. In addition to enhancement mode n-channel and p-channel transistors, the depletion mode n-channel transistor is also available in the technology for flexible circuit design strategy to optimize the circuit performance. The double-poly layers in the HiPAC technology establish the foundation to build Erasable PAL devices. A detailed description of the programmable cell will be given in the next section.

Traditionally in CMOS technology, a p-channel pullup and an n-channel pulldown are used in the basic CMOS inverter logic gate, as depicted in Figure 2. This pullup-pulldown characteristics leads to a very high differential gain and hence a fairly ideal voltage transfer function. Figure 3 illustrates a typical voltage transfer function of a CMOS inverter. Except near the sharp transition region, the output voltage is almost equal to zero or

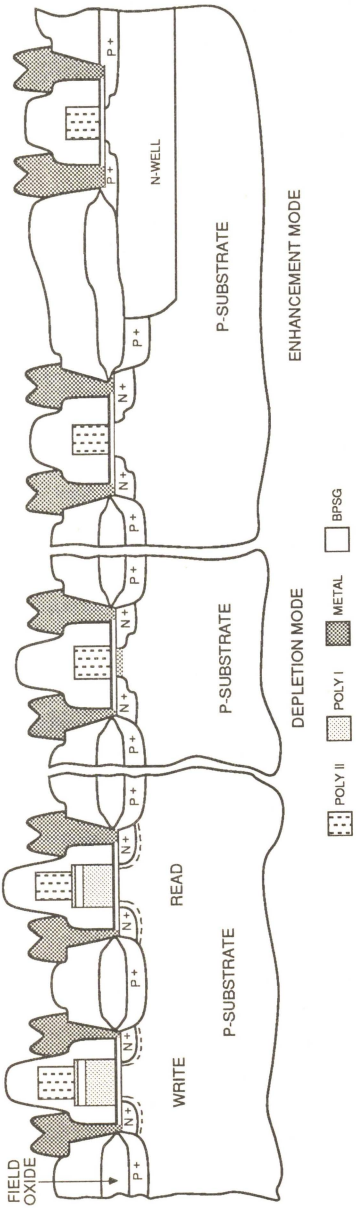



Figure 1. HiPAC Process Cross Section

VDD. These two voltage levels are commonly used to represent logic operating values (0 or 1) for a logic circuit. One key feature of the CMOS inverter is that at either operating condition of these two output logic values, the current flowing through the pullup-pulldown transistor pair is negligible. For this reason, there is almost no power dissipation in the CMOS inverter at either static logic operating levels (zero or VDD). The sharp transition in the voltage transfer function offers another important effect. It makes the maximum allowable logic low value (denoted as VIL) and the minimum allowable logic high value (denoted as VIH) very close to the middle of the voltage swing range (VDD/2). Hence, high noise margin is obtained for the CMOS inverter. All these beneficial properties make the basic CMOS logic inverter nearly an ideal logic element: the output voltage is almost at an ideal logic operating level, and its quiescent power dissipation is also almost zero.

In one version of the advanced HiPAC process, zero-standby power dissipation (with leakage current less than 10 microamps typical and 100 microamps maximum) is attained by using full CMOS inverter gates without back gate bias. The nature of the p-channel pullup and the n-channel pulldown gives large current driving capability, which is about equal in both directions of pullup and pulldown. Therefore, the CMOS inverter features equally fast turn-on and turn-off times. The CMOS inverter logic gate has an additional advantage that it does not suffer from the body effect. This arises from the fact that the body of the n-channel transistor (p-substrate) must connect to the most negative voltage, while the body of the p-channel transistor (N-well) connects to the most positive voltage in the circuit. With such arrangement, the body and the source of both p- and n-channel transistors are shorted together, and the body effect is totally eliminated.

In another version of the process, a back gate bias technique is employed to reduce the parasitic capacitance of the semiconductor junctions, thus enhancing the speed performance of the CMOS circuits. Other advantages obtained from the back gate bias technique include improved field threshold for better isolation as well as better latchup immunity. This second version process requires an on-chip charge pump circuit to generate the back gate bias voltage, and will thus consume somewhat higher power than the first version process. However, it is still only a quarter of the power consumption of equivalent bipolar parts.

The low power dissipation in the CMOS circuits results in lower device junction temperature than their NMOS counterparts. This improves significantly the reliability of CMOS devices, because they will operate with cooler junctions. More devices, and thus more logic functionality, can be integrated onto a single chip. Benefiting from the high device density, CMOS products can grow in complexity to integrate new circuit architectures, and to expand into a new application horizon.

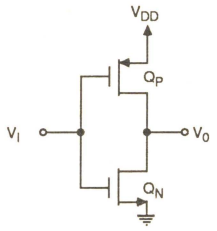



Figure 2. CMOS Inverter

## Programmable Cell Technology and Reliability

HiPAC programmable CMOS devices use the conventional EPROM cell to realize the programming mechanism. An EPROM cell has a layer of polysilicon, called the floating gate, buried within the oxide layers between the regular control gate and the channel region of an MOS transistor. Figure 4 sketches the structure of an EPROM cell. During the programming operation, or the WRITE cycle, a high voltage of 13 to 14 volts is applied to the

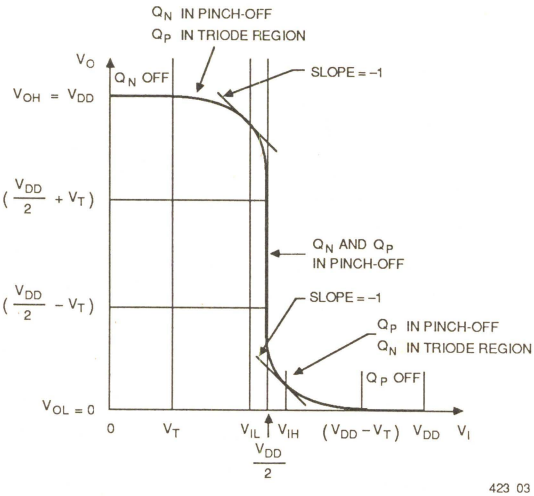



Figure 3. The Voltage Transfer Characteristic of the CMOS Inverter

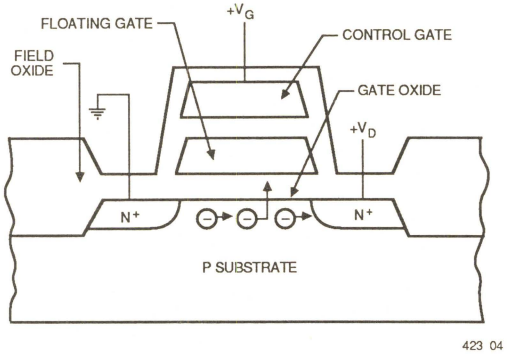



Figure 4. EPROM Ultraviolet Eraseable Cell

control gate and about 12 V to the drain of the floating-gate transistor. The floating gate is capacitively coupled to a positive voltage under the biasing condition. This allows hot electrons conducting in the channel region to be injected into the floating gate, where they become trapped by the potential barrier of 3.1 eV at the polysilicon-oxide interface. The negative electronic charges presented in the buried floating gate tend to shield the channel from the positive voltage on the top control gate. As a consequence, the cell's threshold voltage is raised from Vtu for the unprogrammed (or erased) cell to Vtp for the programmed cell, as depicted in Figure 5. In the normal logic operation, Vtp must be greater than 6 V so that the voltage on the control gate is only high enough to turn on an erased cell, but the programmed cell will remain off.

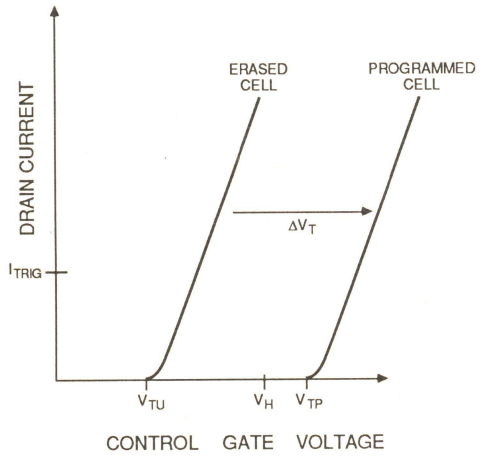
To accomplish the high-speed performance for logic operation, HiPAC has used a two-transistor EPROM cell. A READ transistor is used for sensing the programming condition of the cell in addition to the programming WRITE transistor. The floating gate of the WRITE transistor is shared with the READ transistor. A sense amplifier, connected to the drain of the READ transistor, reads a cell as in an erased state if the drain current is above the amplifier's trigger current ($I_{TRIG}$ in Figure 5) and reads it as in a programmed state otherwise. In general, the greater the read current, the faster the sense amplifier responds. Practically, it is desirable to obtain very high read current without losing the programmability. This is realized with a two-transistor cell, because both the WRITE and READ transistors can be optimized independently to achieve effective programmability as well as high read current to speed up the circuit performance.

Since the logic is programmed as charges stored in the floating gate, the logic pattern is invisible even if a packaged part is opened. This provides security for the users to protect their logic design.

The state of the floating gate, charged or uncharged, is permanent. This is because the floating gate is buried in an extremely high quality oxide, and is electrically isolated with no ground or discharge path. However, with exposure to ultraviolet light of energy greater than 3.1 eV, the trapped charges may be removed and discharge the floating gate. This process is repeatible, and therefore we can take advantage of this feature to conduct functional testing. All cells can be programmed during the manufacturing process and then erased prior to packaging and subsequent shipment. While these cells are programmed, the performance of each individual cell in the programmable array can be fully tested according to the specification. This allows the shipment to the users of parts that have been tested comprehensively to give 100% programming and functional yields.

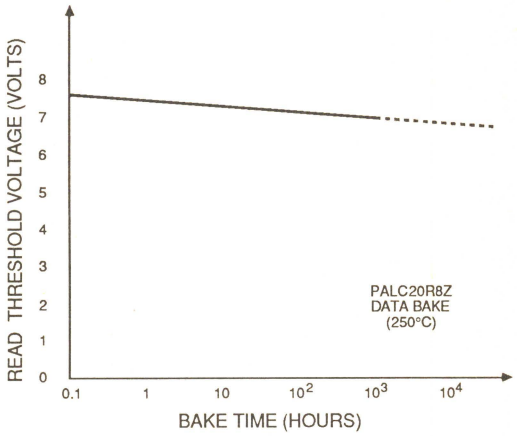
The storage time of the trapped charges in the cell can be characterized by monitoring the change in the threshold voltage of the programmed cell. It has been evaluated through accelerated temperature stress. Figure 6 shows the decrease in read threshold voltage as a function of the total bake time at an elevated temperature of 250° C. The charge retention time is extrapolated to be much greater than 10 years at 125° C. Such impressive charge retention is due to the large potential barrier trapping the stored charges. Contamination and oxide defects may greatly reduce the potential barrier and thereby greatly degrade the charge retention time. With the 100% programming testability, screening of any defective parts is also possible in the manufacturing process. This ensures users the excellent reliability of the parts they will receive.

3




423 05

**Figure 5. Programming of a Cell**




423 06

**Figure 6. Change in Read Threshold vs. Total Bake Time at 250°C**

## Electrostatic Discharge (ESD)

The gate input of an MOS transistor is equivalent to a small, low-leakage capacitor in parallel with a very high resistance, typically $10^{12}$ ohms. Electrostatic charges can readily build up at the gate of the MOS transistor because of this extremely high input impedance. Therefore, all MOS devices are susceptible to electrostatic discharge (ESD) damage if they are not well protected.

To protect the gate oxide against any devastating damage by high levels of ESD, protective circuits are implemented on all CMOS devices. Monolithic Memories has devoted substantial efforts to investigate ESD input protection circuitry. Special attention is paid to design details and layout techniques to give an optimized input protection circuit adapted to each individual CMOS technology. Indeed, every input pin of the CMOS products has been well protected from ESD damage by an appropriate protection circuit.

Figure 7 gives an example of an ESD input protection circuit used in CMOS zero-power ZPAL products. It consists of a thick field oxide transistor (T1), a large-area diode (D1), a distributed diode-resistor (DR) in substrate, and a thin gate oxide transistor (gate-grounded T2) with a low breakdown of approximately 12 V. Large positive input voltages cause T1 to turn on, which will conduct the ESD current to ground. When the voltage between the drain and the source of T2 exceeds 12 V, T2 will break down to further discharge the ESD current. The distributed resistor acts as a current limiter to protect T2 from destruction by too high a current during its breakdown. For large negative input voltages, the diode D1 turns on to dissipate the ESD current.

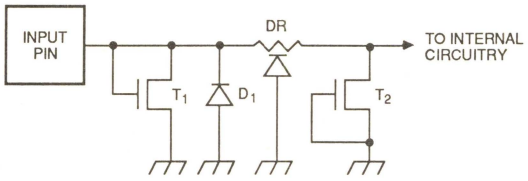
With this ESD protective circuit, any potential destructive discharging current will be dissipated to the ground and thus will not flow into the internal circuitry. Measurements have confirmed that this input protection circuit gives an ESD protection in excess of 2000 V.

For information on latchup, see page 3-160.

## Conclusion

The advanced CMOS HiPAC technology developed at Monolithic Memories has been presented. A family of high performance programmable CMOS devices have been fabricated using this HiPAC technology. They feature fast speed, low power consumption, high density of functionality per unit silicon area, erasable and reprogrammable capability, security of programmed logic pattern, and high reliability. Excellent ESD input protection as well as latchup immunity have been achieved with innovative design techniques and careful layout details.

Thanks to the broad capability of the HiPAC technology, Monolithic Memories is able to offer programmable CMOS products with zero-standby-power consumption and high speed performance. A system designer can now select devices without sacrificing power consumption for speed performance.




423 07

**Figure 7. ESD Input Protection Circuit**

# CMOS EE Technology

In addition to the commonly used bipolar TTL, ECL, and UV-erasable CMOS technologies, programmable logic devices are also manufactured using an advanced CMOS EEPROM-based technology. This technology offers several significant advantages. CMOS allows lower power parts of high complexity. PAL devices based on EE technology can also be reprogrammed electrically, allowing them to be used as a prototyping vehicle. In addition, since the EE-cells can be reprogrammed, these devices can be 100% tested at the factory before being shipped to the customer. CMOS EE-based PAL devices include the AmPALC29M16 and AmPALC29MA16.

This production-tested CMOS process employs state-of-the-art design rules. It uses stepper lithography on all critical levels with a minimum feature size of 1.5 microns. The transistor gate oxide thickness is approximately 300 Å. This advanced process permits volume production of EE-based PAL devices with state-of-the-art speed-power performance. In addition, continued technology enhancements are in development that will result in significantly reduced dimensions and increased packing densities, allowing production of even faster circuits at lower cost.

The EE-cell, which can be electrically erased and reprogrammed, contains a floating gate transistor structure (two layers of polysilicon) with an oxide region of less than 100 Å through which electrons can "tunnel" to either charge or discharge the cell (Figure 1). An additional enhancement transistor has been added in series with the storage cell to prevent leakage in the non-selected discharged cells (as they have a negative threshold) during a charge sensing cycle. This transistor also protects non-selected EE-cells on the same product term during the charge cycle. The tunnel oxide process allows easy manufacturing of ultra-high quality, thermally-grown thin oxide capable of withstanding the high fields associated with the tunneling mechanism.

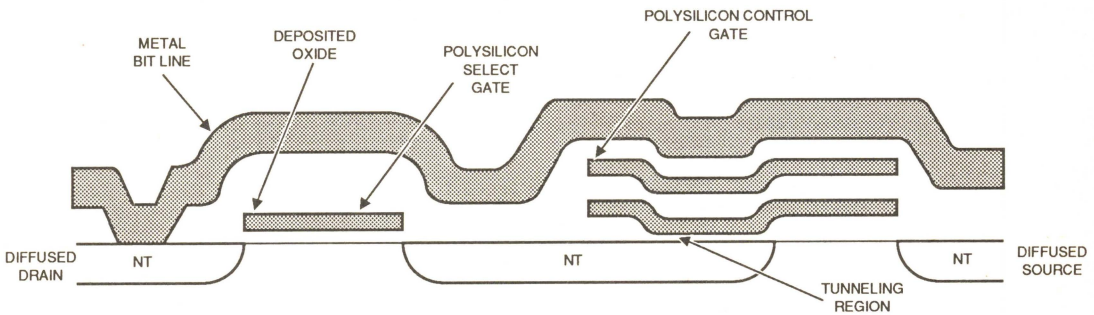
The EE-based process has a proven history of reliability, since it has been used to manufacture EEPROM devices for several years. These devices have been in the field for a long period and have gone through extensive testing at the factory.

Users of devices that are based on EE-cell technology have two concerns about the technology: endurance (number of program or write cycles) and data retention (charge storage from the last time the cell was updated). The endurance issue is of little significance to EE-based PAL device users, as opposed to EEPROM users, since the PAL devices are typically reprogrammed only a few times whereas EEPROMs may be written up to 10,000 times. AMD's process is capable of supporting much higher endurance levels than are specified for PAL devices (100 cycles).

The second concern arises over the leakage of charge from the EE cell over a period of time, thus potentially degrading the device's performance. This issue is resolved by designing the PAL circuits so that performance is guaranteed to the factory specifications under worst case conditions for a minimum of 10 years. If the device is reprogrammed with the same or a different pattern during this period, functionality is assured for another 10 years from that time.

One of the major benefits of EEPROM technology is 100% testability. The EE-based PAL devices can be fully and more easily tested since they are electrically erasable.

With PAL devices available in TTL, ECL and CMOS technologies, the right devices can be supplied for any customer application. Where blazing speed is needed, bipolar is the technology of choice; for low-power high-complexity devices CMOS technology is appropriate. Over the next several years, enhancements will be made to both these technologies allowing production of even faster, lower power and more complex PAL devices. The flexibility to choose technology permits better service to our PLD customers.



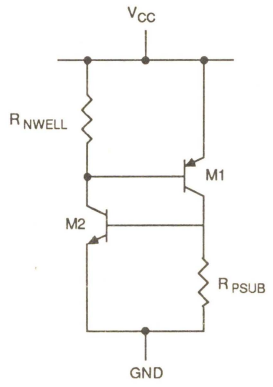

467 01

Figure 1. EE Cell

# Latchup In CMOS Integrated Circuits

## Latchup Circuit

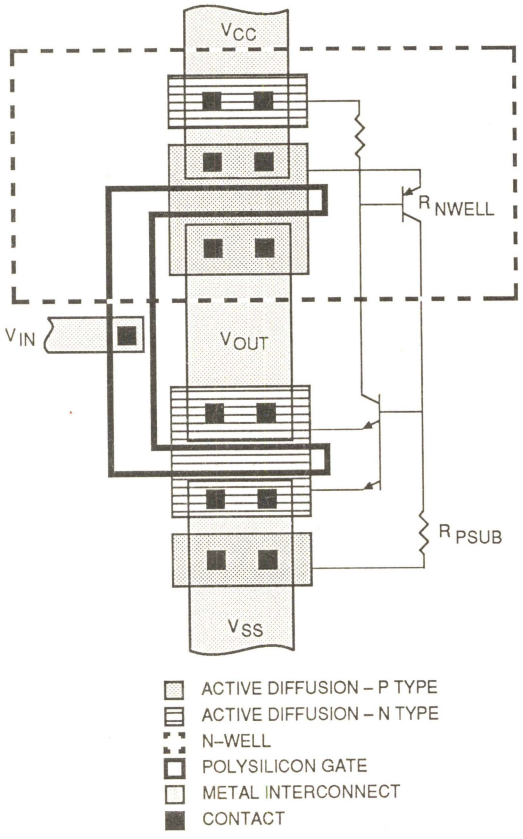
Latchup is caused by an SCR (Silicon Controlled Rectifier) circuit. Fabrication of CMOS integrated circuits with bulk silicon processing creates a parasitic SCR structure. The behavior of this SCR is similar in principle to a true SCR. These structures result from the multiple diffusions needed for the formation of complementary MOS transistors in CMOS processing. The SCR structure consists of a four layer device formed by diffused PNPN regions. These four layers create parasitic bipolar transistors illustrated in Figure 1.




437 01

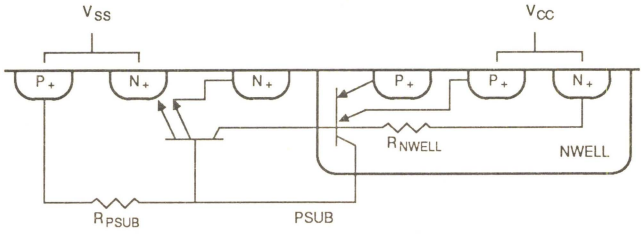
Figure 1

Figure 2a shows a typical CMOS inverter layout with the schematic of the parasitic bipolar SCR structure. Figure 2b is a cross sectional representation of the CMOS inverter, again with the schematic of the bipolar SCR structure.




437 02

Figure 2a




437 03

Figure 2b

Any CMOS diffusion can become part of the parasitic SCR structure, since all of these parts are interconnected through the bulk silicon substrate resistance. Other parasitic resistors shown result from doped regions of the semiconductor. The magnitude to which the resistors resist current flow depends upon geometric size and doping level.

As illustrated in Figure 1, the complementary PNP and NPN transistors are cross-coupled, having common base-collector regions. The vertical PNP device, M1, has its base composed of the N-well diffusion while the emitter and collector are formed from P-type source-drain and substrate regions, respectively. The lateral bipolar transistor, M2, base is the P substrate with emitter and collector junctions formed from N-type source-drain and N-well diffusions, respectively.

### Latch-Up Conditions

Under normal bias conditions the SCR conducts only leakage current and the SCR structure is in the blocking state. However, as current flows across any of the parasitic resistors, a voltage drop is developed, turning on the parasitic bipolar base-emitter junction. The forward bias condition of this junction allows collector current to flow in the bipolar transistor. This collector current flows across the base-emitter resistor of the complementary bipolar transistor, creating a voltage sufficient to turn on the transistor.

A regenerative loop is now created between the complementary bipolar transistors such that current conduction becomes self-sustaining. Even after removal of the stimulus that triggered this action, the current conduction can continue. This region of operation is a high-current, low-resistance condition characteristic of a four layer PNPN structure. This is referred to as latchup. Once initiated, the excessive latchup current can permanently damage an integrated circuit by fusing metal lines or destroying junctions.

## Causes Of Latchup

Latchup may be initiated in numerous ways. Just the critical causes frequently encountered in a system environment will be discussed. These include power up, supply overvoltage, and overshoot/undershoot at device pins.

### Power-Up

Caution must be exercised when powering up CMOS ICs to avoid driving device pins before the supply voltage has been applied to the circuit. Placing a device or board in a "hot socket" will create this situation. When subjected to hot socket insertion, voltage conditions at the device pins are uncertain such that the input diodes may be forward biased. Forward biasing the input diodes with a delayed or uncontrolled application of VCC could cause the device to latch up. Monolithic Memories' CMOS circuits have substantial immunity to hot-socket power up, but since this condition is uncertain, and difficult to characterize, test, and guarantee, it should be avoided.

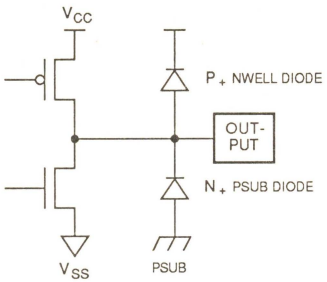
### Supply Overvoltage

Supply levels exceeding the absolute maximum rating can cause a CMOS circuit to latch up. Elevated supply voltage may cause internal junctions to break down, producing substrate current capable of triggering latchup. Latchup is just one of the reasons overvoltage should be avoided; other undesirable effects may result from this.

### Overshoot/Undershoot

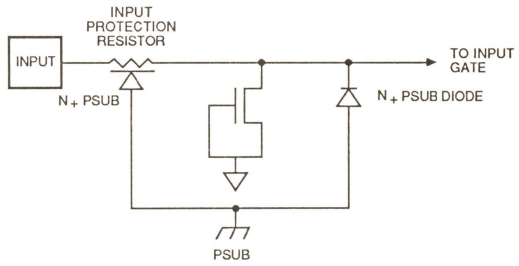
Generally the I/O pins experience the noisiest electrical environment. Fast switching signals with a large capacitive load may overshoot, creating a transient forward bias condition at the I/O junction. These junction diodes are illustrated in Figures 3 and 4. Typically this is where latchup is most likely to be induced. Proper design of the input and output buffers is essential to minimize the risk of latchup due to overshoot.





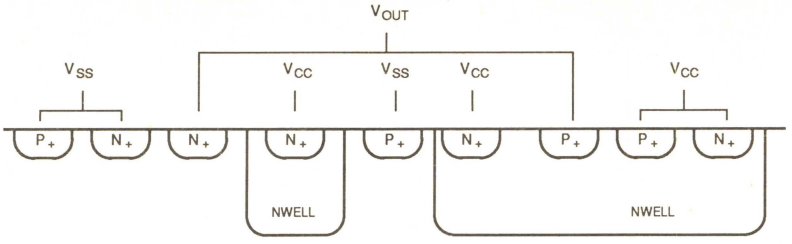

437 04

Figure 3




437 05

Figure 4




437 06

Figure 5

## Reducing Latchup Susceptibility

Numerous methods have been proposed to reduce the susceptibility of CMOS circuits to latchup. Some of these include gold doping, conductive substrates with epitaxy, substrate bias generators, guard rings, and geometric spacing. Many of the above methods may compromise the performance of the active circuit components while limiting the parasitic SCR. The extreme immunity some of the approaches provide may be in excess of what is needed for a reasonable system environment. It would then be difficult to justify the complexity and added cost to produce unnecessary immunity to latchup.

Special attention is placed in the design and layout of the input and output buffers to reduce the susceptibility to latchup induced by overshoot. Diffused guard rings connected to VCC and ground are placed around I/O diffusions to collect injected charge. The charge is passed directly to the power supply rather than to active regions of the parasitic bipolar transistor. These guard rings, as shown in Figure 5, also reduce substrate current as well as effective substrate resistance, making bipolar turn-on more difficult. As an added precaution, diffusion space at the I/O is relaxed to reduce bipolar transistor gain.

## Latchup Prevention in Low-Power CMOS PAL Devices

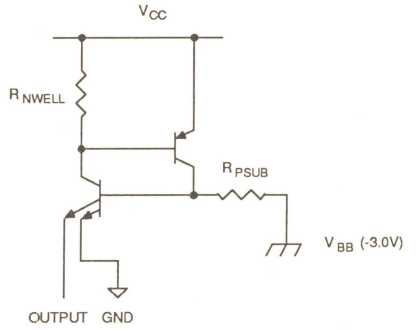
Latchup in CMOS ICs is triggered by developing a forward bias condition across the base-emitter junctions of the parasitic NPNP SCR structure. As discussed earlier, this is most likely to occur at inputs and I/O pins as a result of system overshoot during switching transitions. To increase the undershoot latchup noise margin available at inputs and I/Os, a technique to bias the substrate voltage to a negative level is employed.

Incorporated into the design is a substrate pump, or bias generator. The bias generator capacitively couples the substrate to a negative potential (VBB), providing an additional 3 V undershoot margin prior to forward biasing the N+/psub diode (Figure 6). No additional supply voltages are necessary to hold the substrate negative; the on-chip substrate pump accomplishes this with the standard +5 volt supply. Impedance of the bias generator is determined by several design parameters.

Operation of the substrate bias generator continuously draws several milliamps supply current. The power is dissipated by a free running oscillator to charge and discharge the substrate pump capacitors. This level of power dissipation is within an acceptable range for use in quarter and half power high speed CMOS PAL devices. For this family of devices the array power dissipation is many times greater than the power consumed by the substrate bias generator and can therefore be tolerated.

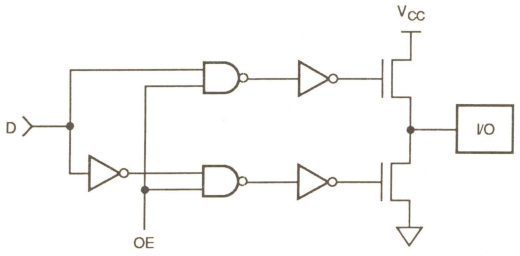
Output drivers in Monolithic Memories high-speed CMOS PAL devices are exclusively N channel as shown in Figure 7. Removal of the P+ region eliminates the overshoot latchup sensitivity found in traditional CMOS outputs. The result is a CMOS circuit tolerant to overshoot. Proper design of the NMOS pull up configuration allows adequate VOH level for TTL compatibility.

When used in conjunction with a pumped (–3 V) substrate, latchup cannot be initiated with realistic system overshoot or undershoot. These design techniques have made the CMOS PAL devices essentially latchup free, even up to 200 mA sink/source current at inputs and I/Os.




437 07

Figure 6




437 08

Figure 7

## Latchup Prevention in Zero-Power CMOS PAL Devices

Another strategy must be taken for zero power ZPAL logic devices since these products must consume no power when in the quiescent standby operating mode. This restriction eliminates the possibility of an on-chip bias generator. Additionally, ZPAL devices require CMOS VOH levels, placing further constraint on output buffer design.

Common to many CMOS output buffer circuits is a complementary PMOS and NMOS output driver. A P-channel pull-up transistor allows the VOH level to reach VCC; however, placement of the P-channel transistor at the output also creates a large P+ diffusion region at the I/O. This diffusion acts as an injector for carriers during positive overshoot at the I/O pin. Under worst-case overshoot conditions, the injected charge may be large, sufficient to turn-on (VCC +0.6) the P+/N-well diode to the extent that the four layer SCR structure will latch up. Once latchup is initiated the device will most likely be permanently destroyed.

Proper placement of guard rings and dummy collectors around the P+ diffusion at the I/O will only increase the amount of current necessary to trigger latchup; ultimately the SCR can be turned on even with these layout precautions. Avoiding placing the P channel device at the I/O increases the circuit's immunity to latchup when compared to a traditional full CMOS output utilizing extensive layout safeguards.

## Testing For Latchup

Monolithic Memories characterizes the latchup sensitivity of its devices before they are released to the market. Testing is done in such a way as to completely cover every possible latchup condition, including VCC overvoltage, pin overcurrent, and pin overvoltage.

### VCC Overvoltage Test

The VCC overvoltage test is applied to all power (VCC) pins. The test is performed at the highest guaranteed operating temperature of the device. All inputs and I/Os acting as inputs are tied to ground or VCC depending on the device logic, and outputs and I/Os acting as outputs are floating (open).

VCC max is applied to the VCC pin. A positive high voltage pulse is then applied to the VCC pin and returned to VCC max. The occurence of latchup is detected if the voltage across the device is less than VCC max, and the current through the device is greater than the normal DC operating current.

### Pin Overcurrent Test

The pin overcurrent test is performed on every output, I/O pin, and non-current-limited input pin. Non-current-limited inputs are inputs which present a diode-like (or otherwise "infinite") current characteristic for input voltages in the range (GND – 5 V) < Vin < (VCC + 5 V).

The pin overcurrent test is performed at the highest guaranteed operating temperature of the device. Input pins and I/O pins acting as inputs (which are not under test) are tied to ground or VCC depending on the device logic, and outputs and I/Os acting as outputs should be floating (open). VCC max is applied to the VCC pin.

One pin is tested at a time. A three-state output under test should be disabled. A non-three-state output type under test should be a logic High when applying a positive current and a logic Low when applying a negative current. An I/O pin should be placed into the input mode.

A high current pulse is then applied to the pin under test. The magnitude of the pulse is stepped until latchup is induced. Both positive and negative currents are tested. Latchup is observed as described previously. The sensitivity of the device is the worst case sensitivity found on any pin of the device.

### Pin Overvoltage Test

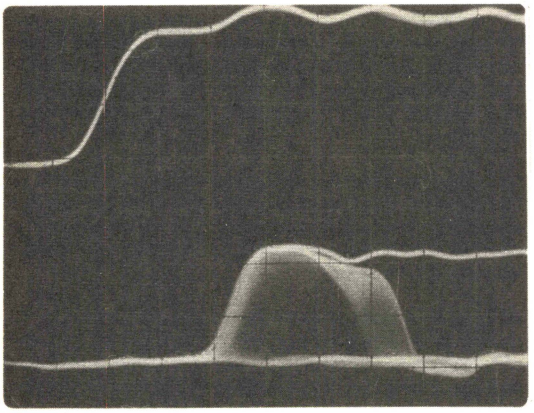
The pin overvoltage test is performed on current-limited inputs. Current-limited inputs are inputs which present a resistor-like (or otherwise "limited") current characteristic for input voltages in the range (GND – 5 V) < Vin < (VCC + 5 V).

The pin overvoltage test is performed at the highest guaranteed operating temperature of the device. Input pins and I/O pins acting as inputs (which are not under test) are tied to ground or VCC depending on the device logic, and outputs and I/Os acting as outputs are floating (open). VCC max is applied to the VCC pin.

One pin is tested at a time. Both positive and negative voltage pulses are applied to the pin under test. Latchup is observed as described previously. The sensitivity of the device is the worst-case sensitivity found on any pin of the device.

# Metastability

## A Study of the Anomalous Behavior of Synchronizer Circuits



### INTRODUCTION

This article will summarize the results of the studies performed on synchronizer circuits. The information presented may be used by system designers to gain insight into the anomalous behavior of edge-triggered flip-flops. Understanding flip-flop behavior and applying some simple design practices can result in an increased reliability of any system.

### METASTABILITY

In the digital world a bit represents the fundamental unit of measure. The output state of any digital device is either "HIGH" (a voltage level above VIH) or "LOW" (a voltage level below VIL) as shown in figure 2. Under the proper operating conditions the register in figure 1 outputs a HIGH or a LOW on the rising edge of the clock within a nominal delay called the "clock to out" delay. If the setup and hold times are violated the register has a small probability of entering a third region of operation called the "metastable" state. Metastable is a Greek word meaning "in between" and it is a state between HIGH and LOW. Even though most synchronizers snap out of metastability in a short period of time, theoretically this state can persist indefinitely. Some of the registers built from older technologies had metastable states which lasted as long as a few microseconds. When the output of a device goes into metastability the clock to out delay will be grossly affected. This may alter the system's worst case propagation delay and potentially lead to a system crash!

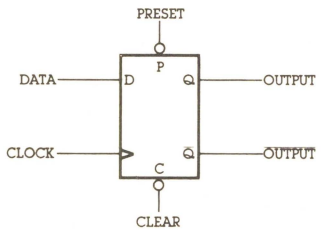



Figure 1

### SYNCHRONIZERS

The design of a synchronous digital system is based on the assumption that the maximum propagation delay of a flip-flop and any other gates are known. A digital system is free of hazardous race conditions and timing anomalies if the maximum propagation delay in the system does not exceed the clock's period. In systems where an asynchronous input is interfaced with a clocked device such as a flip-flop, the maximum specified propagation delay of this device may no longer be valid if certain electrical parameters are violated. Computer peripherals, an operator's keyboard, or two independently clocked subsystems are instances where there is a possibility of interfacing an asynchronous input which will violate the synchronizer's electrical parameters.

A popular device typically used in synchronized systems is the edge-triggered register shown in figure 1. The edge-triggered register will properly synchronize the incoming data to the system's clock as long as its operating conditions are satisfied. Table 1 summarizes these specifications for Monolithic Memories Inc's (MMI) 74LS374 register. It is difficult to guarantee setup and hold time requirements when the data is asynchronously interfaced to a register. The violation of setup or hold time in a register has a probability of initiating a misbehavior termed "Metastability."

| SYMBOL | PARAMETER | COMMERCIAL MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply Voltage | 4.5 | 5 | 5.5 | V |
| $T_A$ | Operating free air temp. | 0 | | 75 | °C |
| $t_w$ | Width of clock | 15 | | | ns |
| $t_{su}$ | Setup time | 20 | | | ns |
| $t_h$ | Hold time | 0 | | | ns |

Table 1

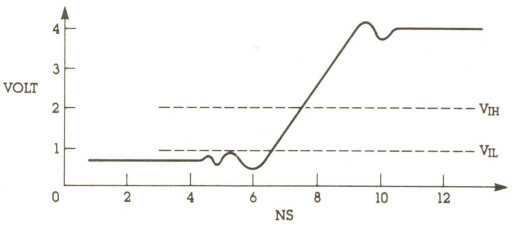



Figure 2

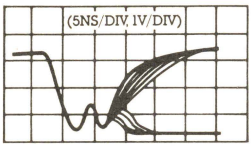
The diagrams in figure 3 illustrate some examples of waveforms in the metastable condition. From the waveforms it is evident that the outputs are distorted under metastable conditions. Figure 3d shows the output of a typical 74LS374 register manufactured by Monolithic Memories. Monolithic Memories family of bipolar devices exhibit superior metastable hardened performance due to their high speed bipolar technology and advance Schottky TTL circuit design techniques. Most of these devices typically snap out of metastability in a flashing 15 nanoseconds.

### WHY THE SYNCHRONIZER FAILS

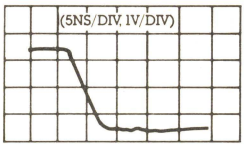
Before attempting to explain how the synchronizer's internal circuitry fails let's take a look at an interesting problem.

PROBLEM: In the SR type latch shown in figure 4 what happens if the set (S) and the reset (R) inputs are simultaneously raised from a LOW voltage level to a HIGH level?

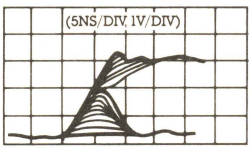
ANSWER: The outputs will be in a stable state of HIGH prior to the RS transition and will quickly oscillate to a final steady state of either HIGH or LOW (see figure 3a). To demonstrate this result the reader is encouraged to do this excercise either mentally or to actually build the circuit and view the output on the oscilloscope.



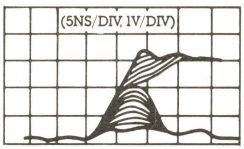

(a) CROSS-TIED NAND GATES




(b) REGISTER WITH NORMAL BEHAVIOR




(c) MONOLITHIC MEMORIES INC'S PAL16R4A




(d) MONOLITHIC MEMORIES INC'S 74LS374

Figure 3

Clock driven master-slave flip-flops contain the same type of cross tied RS latch within their internal circuitry. The NAND gate equivalent of the master-slave D type flip-flop is shown in figure 5. The gates circled in this figure can potentially behave similar to the above problem. If the clock and data are triggered within a specific window of one another the output may have an oscillatory behavior before settling down.

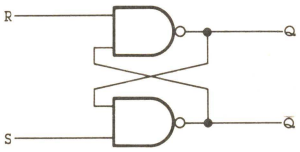
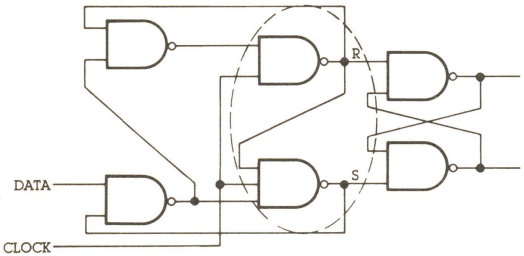



Figure 4




Cross tied RS latch structure is seen in the master-slave edge triggered flip-flop.

Figure 5

## METASTABLE DETECTOR

This section will show how to characterize the behavior of an edge triggered flip-flop with an asynchronous data interface. If the setup and hold times of the flip-flop are satisfied the output behaves properly (figure 6a). One of the four possible events below can take place if the flip-flop goes metastable:

1) The output starts to make a transition but snaps back to its original state (figure 6b).

2) The output makes a complete transition but the maximum propagation delay of the device is exceeded (figure 6c).

3) The output starts oscillating and retains its present state (figure 6d).

4) The output oscillates to a new state (figure 6e).

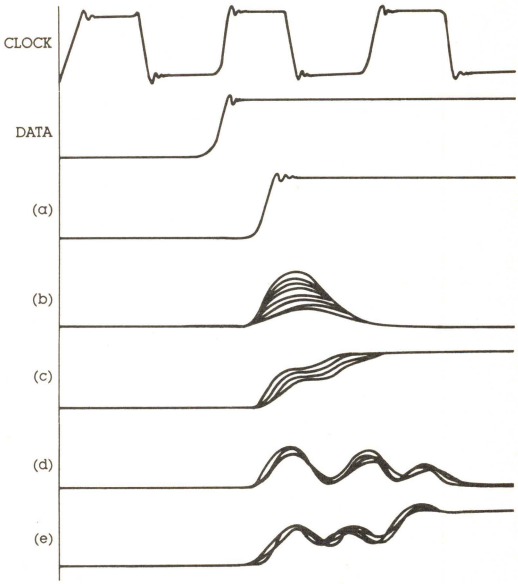



Figure 6

The circuit shown in figure 7 is used to obtain experimental results of a metastable device. The circuit can detect and count the number of events of metastability. The device under test (DUT) is forced into metastability by repeatedly sweeping the edges of the data past the rising edges of the clock. The modulation of the data is possible by using a comparator device (U1) along with an external sawtooth waveform. Thousands of transitions are created within the setup and hold time window of the DUT. Sweeping the data edges past the low to high clock transitions simulates an asynchonous input and increases the probability of getting a metastable failure on the output ($\overline{Q}$) of the DUT.

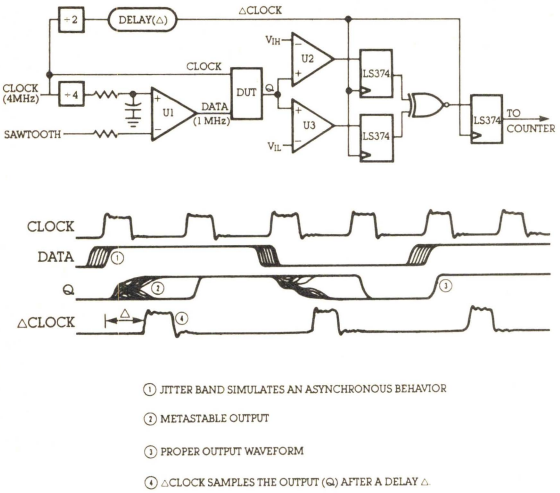



Figure 7

If the output of the device goes into metastability it will be detected by the comparator pair (U2) and (U3). The comparators will have complementary outputs if the output (Q) of DUT is anywhere between VIH and VIL. The outputs of the comparators are latched by a delayed version of the clock (ΔClock). The EXCLUSIVE-NOR gate followed by the register signal the event of metastability to an external counter.

The variable delay (Δ) between the two clocks will sample the output at various locations on the time axis. As this delay is varied the event of metastability is sampled and counted at these locations by our circuit. Therefore the output of our circuit measures the rate of metastability versus time delay. The real behavior of a metastable output can thus be effectively characterized with this scheme, that is, we can determine the length of time a metastable condition will persist and the density distribution of the metastable event.

Three 74374 devices and four PAL devices are used in this experiment. The plots of metastable failure versus time are shown in figures 8a,b. The next section will discuss in detail the characteristics of these plots.

## EXPERIMENTAL RESULTS

Various graphs of metastability failure rate versus delay time are illustrated in figure 8. We can conclude from these graphs that the rate of metastability failure decreases as the sample clock (ΔCLOCK) moves farther and farther away from the DUT clock. The pictures shown in figure 9 have captured repeated events of metastability on the oscilloscope.

Let's take a closer look at one of the graphs to examine the behavior of the device. The PAL16R4A-4 device exhibits one count per second if the delay (Δ) is 60 nanoseconds. As the delay (Δ) is decreased, the rate increases exponentially until the delay equals 32 ns at which point the rate flattens out and remains fixed. The 32 ns forms the knee of our graph and will be referred to as Δo. The rate will remain constant if the delay (Δ) is decreased past the knee of our graph. Further reduction in the delay will place the sampling clock's rising edge prior to data transitions and thus the error rate vanishes to zero. The time at which the rate goes to zero is marked with an (X) on the graphs. By using this time (X), and another location on the graph such as the time where only one error per second occurs, we can associate an approximate range of metastability for different devices. This range of metastability is referred to as the "mean time to snap out of metastability". From the graph it is evident that the mean time to snap out of metastability for the PAL16R4A-4 logic circuit is the difference between 60 ns and 25 ns which is 35 ns.

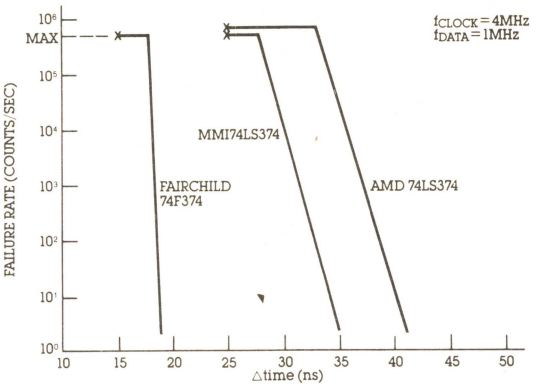



Figure 8a

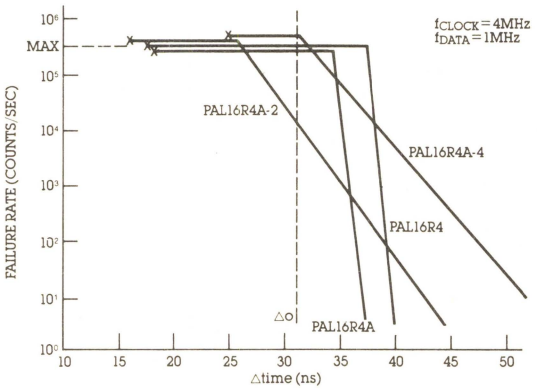



Figure 8b

All of the graphs illustrated can be quantified by an equation of the form:

$$\log \text{FAILURE} = \log \text{MAX} - b(\Delta - \Delta o)$$

Since a natural logarithm is a constant multiple of base 10 logarithm we can rewrite the above equation as:

$$a \cdot \ln \text{FAILURE} = a \cdot \ln \text{MAX} - b\,(\Delta - \Delta o)$$

In the above equation the MAX value is representative of the maximum metastability failure rate in our device. This MAX value is closely related to the frequency at which a metastable condition may occur in our device. The frequency at which metastability occurs is simply a constant multiple of the product of CLOCK and DATA frequency.

$$\text{MAX} = K1 \cdot f_{CLOCK} \cdot f_{DATA}$$

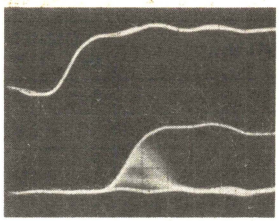
Substituting this in our original equation we get:

$$a \cdot \ln \text{FAILURE} = a \cdot \ln (K1 \cdot f_{CLOCK} \cdot f_{DATA}) - b(\Delta - \Delta o)$$

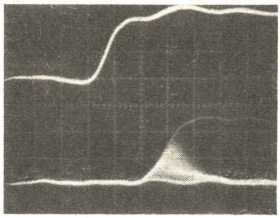
$$\ln \text{FAILURE} = \ln (K1 \cdot f_{CLOCK} \cdot f_{DATA}) - b/a(\Delta - \Delta o)$$

$$\text{FAILURE} = (K1 \cdot f_{CLOCK} \cdot f_{DATA})\, e^{-k2\,(\Delta - \Delta o)}$$

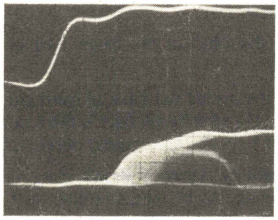
PAL16R4



PAL16R4A



PAL16R4A-4



PAL16R4A-2

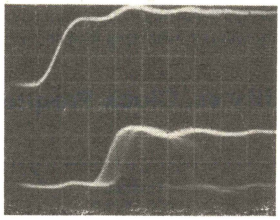


Figure 9 (2v/DIV, 5ns/DIV)

Table 2 gives the three important parameters which can be used by system designers to fully characterize the metastable behavior of the mentioned devices. These parameters can be obtained for different devices by duplicating this experiment. An example is given below to show how the information on table 2 may help the designer in the design of asynchronous systems.

| MANUFACTURER | DEVICE | $K_1$ (Sec) | $K_2$ ($ns^{-2}$) | $\triangle o$ (ns) |
|---|---|---|---|---|
| MMI | PAL16R4 | 1 x 10 $^7$ | 4.3 | 37 |
| | PAL16R4A | 1 x 10 $^7$ | 4.3 | 34.5 |
| | PAL16R4A-2 | 1 x 10 $^7$ | .64 | 25 |
| | PAL16R4A-4 | 1 x 10 $^7$ | .5 | 31 |
| | 74LS374 | 2 x 10 $^7$ | 1.8 | 27.5 |
| AMD | 74LS374 | 2 x 10 $^7$ | 2.0 | 34.5 |
| FAIRCHILD | 74F374 | 2 x 10 $^7$ | 11.5 | 17.5 |

Table 2

EXAMPLE

For the hardware implementation in figure 10 determine the maximum clock frequency to give a typical error rate of one failure per year. We must choose the minimum period to give an error rate of less than

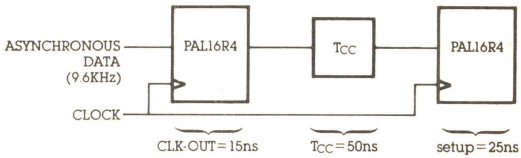



Figure 10

one failure per year. From this result we can determine the maximum clock frequency. The time $\Delta$ in the equation below will determine the distance between clock edges. We must determine $\Delta$ from the equation by numerical extrapolation. The system clock's period can be represented as ($\Delta$ + Tcc + setup), or plugging in the numbers it is $\Delta$+75.

$$\text{FAILURE} = (K1 \cdot f_{CLOCK} \cdot f_{DATA})\, e^{-K2(\Delta - \Delta o)}$$

and plugging in the appropriate values we have:

$$3.2EE-8 = [(1EE-7)(1/(\Delta+75ns))(9600)]\, e^{-[(4.3)(\Delta-37)]}$$

Solving for $\Delta$, we see that it is approximately 43 nanoseconds. The system period is thus seen to be the sum of 43ns and 75ns or 118ns. The maximum clock frequency is the inverse of the period or approximately 8 MHz.

CONCLUSION

Synchronization of two independent pulse trains is possible through the use of edge triggered registers. The electrical characteristics of the flip-flop are affected when the setup and hold times of the device are violated. This misbehavior is termed "metastability" and its probability of occurrence can be derived for a given system. The factors which affect this probability and the length of time which a metastable condition persists are influenced by the technology of the device as well as by the circuit design techniques.

An important fact which needs to be stressed is that even if a register's output goes metastable, the system may not necessarily fail if the register snaps out in time to satisfy the system's worst case timing requirement. The following design practices are suggested when using synchronizers:

Try to minimize the number of locations where asynchronous signals enter your system.

Clocking the asynchronous inputs through two pipelined registers can greatly reduce the error rate.

Use a single clock within your local system environment. For multiple system clocks, derive all the clock signals from a single source to assure synchronization between different devices within the system.

When analyzing the worst case timing of your system, add the time to snap out of metastability to any register in an asynchronous data path.

A single PAL® with registers can be your best choice for state machine analysis of asynchronous events. As the registers have virtually identical setup times, the simultaneous observation of a metastable event by different register states are likely to be the same. Contrasted to a distributed system of observing register states with different setup times, the PAL system of register states with identical setup times is a superior synchronizer.

Avoid edge sensitive devices on the output paths of the registers which have asynchronous inputs. The glitch created when the synchronizer goes metastable is enough to trigger the edge sensitive device. The use of level sensitive devices is generally a better design practice.

PAL devices can be effective synchronizers where various registering schemes are easily implemented.

3

## Metastability Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| p | Poisson process rate | | 0.85 | 1.05 | | $ns^{-1}$ |
| k | MTBF constant | | | 0.8 | 1.0 | $\mu s^{-1}$ |
| $t_{MET}$ | Minimum recovery time in asynchronous mode | MTBF = 10 years $f_d = (1/3)f$ d = 3 | | 20 | 30 | ns |
| $f_{MET}$ | Maximum frequency in asynchronous mode | MTBF = 10 years $f_d = (1/3)f$ d = 3 | 21 | 26 | | MHz |

## Metastability

Metastability is a condition which can occur in any latch or flip-flop if the minimum setup or hold times are violated. In most cases, the flip-flop will either react to the input or remain in its current state, both of which are stable results. The flip-flop can also reach an "in-between" condition called the metastable state, which is stable only if there is no noise in the system and the flip-flop is perfectly balanced. This metastable condition lasts until the flip-flop falls into one of its two stable states, which can take longer than the normal response time.

The PAL16R8-10 Series exhibits better metastability characteristics than most other registered devices. It is less likely to enter the metastable state and recovers faster to a stable state. As a result, the PAL16R8-10 Series can make an excellent synchronizer circuit, and the metastability characteristics have been specified for designs in which the setup and hold times may not always be met.

## Definition of Variables

**MTBF (Mean Time Between Failures):** the average time between metastable occurrences that cause a violation of the device specifications. Metastability characteristics are calculated at an arbitrary MTBF of 10 years for the convenience of the user.

**p (Poisson process rate):** experimentally calculated factor which determines the slope of the curve of probability of failure.

**k (MTBF constant):** experimentally calculated factor which determines the magnitude of the curve of probability of failure.

**tsu (setup time):** the specified minimum time interval allowed between the application of a data signal at a specified device input pin (pin 9 on the device under test) and a subsequent clock transition. For the PAL16R8-10 Series, tsu is 10 nanoseconds.

**tCLK (clock to output time):** the specified maximum time interval between a clock transition and the availability of valid signals at an output pin. For the PAL16R8-10 Series, tCLK is 8 nanoseconds.

**fMAX (maximum frequency):** specified maximum frequency for the device under test. Calculated as 1/(tsu + tCLK). For the PAL16R8-10 Series, this calculates to 55.5 Megahertz.

**f (clock frequency):** actual clock frequency for the device under test.

**$f_d$ (data frequency):** actual data frequency for a specified input to the device under test.

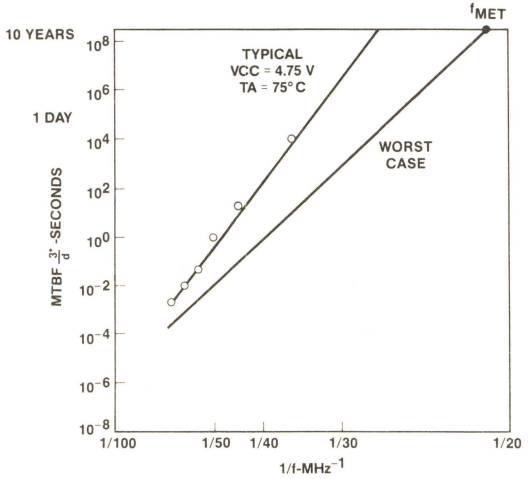
**d (data ratio):** the ratio of the clock frequency to the data frequency ($f/f_d$).

**t (time delay):** the additional time allowed per period beyond that required by the specifications. t is the actual time between clock transitions beyond the required period of (tsu +tCLK).

**t MET (metastability recovery time):** minimum t required to guarantee recovery from metastability, with specified test conditions.

**f MET (metastability frequency):** maximum f clock frequency to limit metastability failures, with specified test conditions.

## Metastability vs. Clock Frequency




* Normalized to d = 3; multiply by 3/d for other data frequencies.

## Metastability Equations

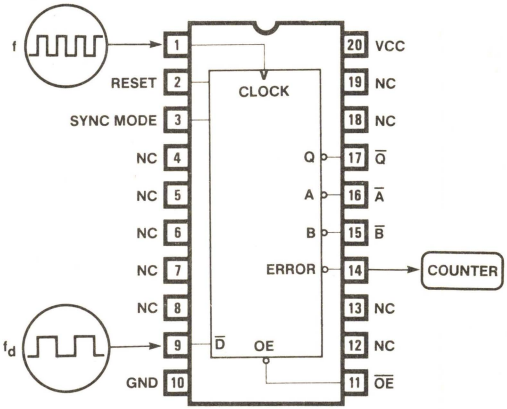
MTBF = k (d/3) $(1/f)^2$ $e^{(p/f)}$

f MAX = 1/(tsu + tCLK)

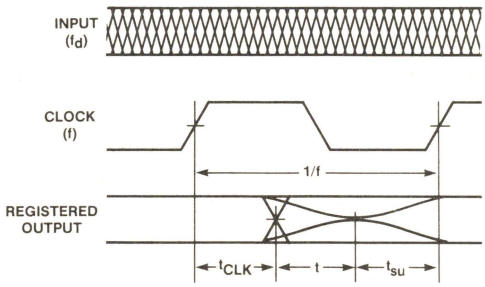
f = 1/(tsu + tCLK + t)

f = d ($f_d$)

## Metastability Test Circuit




## Metastability Waveforms




## Metastability Test Pattern File

```
CHIP Metastability_Test PAL16R4

CLOCK RESET SYNC_MODE NC NC NC NC NC /D GND
/OE NC NC /ERROR /B /A /Q NC NC VCC

EQUATIONS

Q      := /Q* SYNC_MODE    ;TOGGLE SYNCHRONOUS INPUT (TESTS f MAX)
        +  D*/SYNC_MODE    ;TOGGLE ASYNCHRONOUS INPUT (TESTS META.)

A      :=  A*/Q            ;HOLD A (IF NOT ERROR)
        + /A* Q            ;TOGGLE A (IF NOT ERROR)
        +  ERROR           ;SET A IF ERROR

B      :=  B*/Q*/ERROR     ;HOLD B IF NOT ERROR, OR RESET
        + /B* Q*/ERROR     ;TOGGLE B IF NOT ERROR, OR RESET

ERROR := /A*/B             ;COMPARE A AND B,
        +  A* B            ; ERROR GOES HIGH IF A EQUALS B
        +  RESET           ;INITIALIZE A AND B TO OPPOSITE PHASES
```

Note:

Metastability characteristics were experimentally determined using the method shown, and are not guaranteed.

# Surface Mount Technology

## Monolithic Memories

Monolithic Memories, Inc. (MMI) has long been a leader in integrated circuit packaging technology. MMI was one of the first to offer integrated circuits in 24-pin, 300-mil wide SKINNYDIP® packages, and pioneered the use of Pin Grid Arrays.

Now Monolithic Memories again proves its commitment to packaging innovation by introducing commercial products not only in the surface mount Plastic Leaded Chip Carrier (PLCC) package, but also in the Small Outline (SO) Gull-wing package.

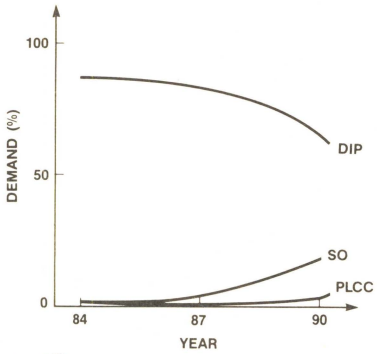
Surface Mount Technology (SMT) is creating exciting changes in the design and construction of Printed Circuit Boards (PCBs). Surface mount boards have significantly higher density, lower cost, and higher reliability than boards manufactured using DIP packages.

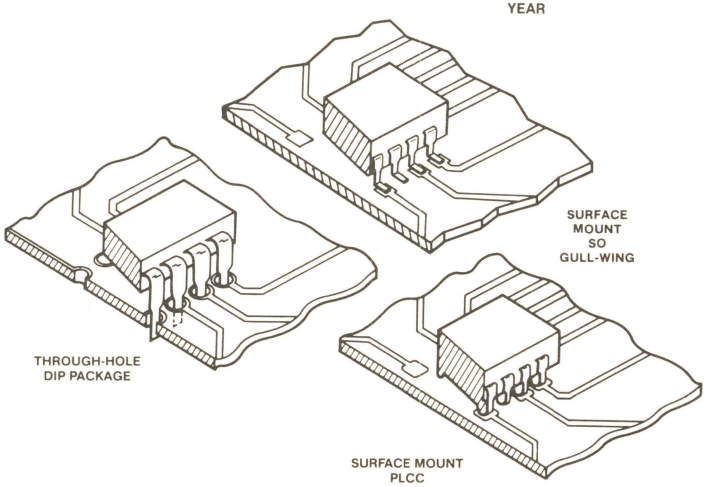
## What is Surface Mount Technology?

Surface mount circuit boards are assembled with components bonded to metal pads on the board surface, instead of being inserted into through-holes like conventional DIP packages. Surface mount components can be mounted on both sides of a circuit board, DOUBLING the functional density. In addition to packages for integrated circuits, surface mount versions have been developed for discrete devices, such as resistors and capacitors, and also for single transistors.

## Development of SMT

Surface mount technology was first developed and employed in the hybrid industry over twenty years ago. Since then surface mount packages have gained increasing acceptance. Benefits of switching to SMT include manufacturing cost reduction, an overall reduction in the size and weight of device packages, and improvements in circuit performance. It is estimated that by 1990 40% of components shipped will be in surface mount packages.




**Cutaway View Comparison of Surface Mount and Through-hole components**

## Advantages of Surface Mounting

Surface mount technology offers increased performance, reliability and workability at lower cost.

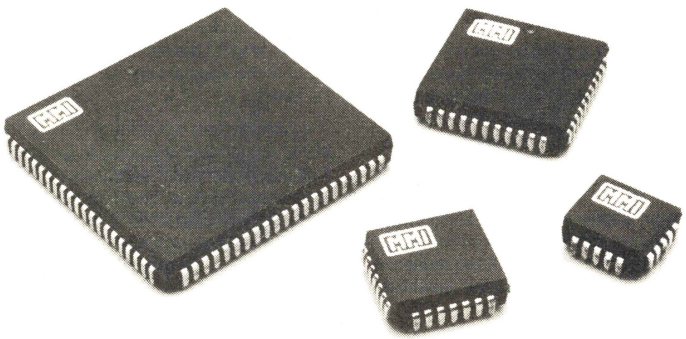
### Cost

- Automated surface mount assembly dramatically increases board throughput and lowers the cost per board:
  - Less time per board
  - Less labor per board
  - Less potential for error in board fabrication
  - More reproducible process
  - Higher quality control
- Surface mount reduces the number or size of boards required, due to the reduced size of the components.
- Boards are less expensive because fewer through holes are needed
- Higher board yields can be achieved with SMT through improved component reliability
- Expensive, special substrates are not needed for surface mount packages

### Reliability

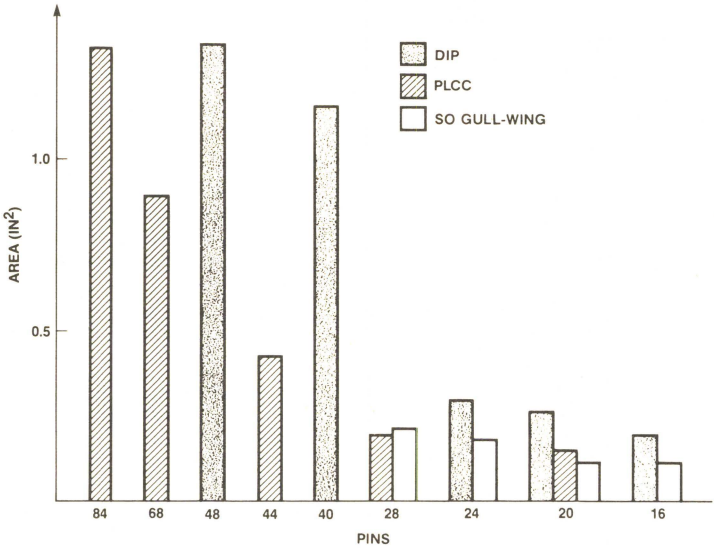
- Monolithic Memories' surface mount packages are manufactured using the same dependable materials and process as the standard molded DIP package
- PLCCs and SOs have been shown to be equal or superior in terms of reliability to DIP packages
- Solder joint reliability is improved through the use of vapor phase reflow soldering. Where conventional DIP boards need to pass over a heated coil or a molten solder wave, vapor phase reflow soldering can control more tightly the high temperatures required to melt the solder
- PLCCs have superior stress and vibration performance
- Surface mount packages are rugged and durable





**Performance**

- PLCCs and SOs are smaller than DIP packages up to 75% which increases the board functional density by as much as 4:1.
- PLCCs and SOs are lighter than DIP packages up to 90% which reduces overall board weight.
- PLCCs and SOs can be mounted on two-sided PCBs which increases the board density even further.
- PLCCs have improved electrical performance due to shorter lead lengths.
- PLCCs have decreased susceptibility to noise.
- The PLCC's "J-Hook" leads absorb thermal and mechanical stress.




## Board Testing and Repair

As surface mount components increase the density and performance of PCBs they also make board testing and repair more complex. New techniques are required not only for product design and assembly, but also for testing and repair functions.

### Testing

Testing begins long before a product has been fully assembled. Bare PCBs are often tested for continuity and isolation before components are soldered on them. Spring-loaded (pogo) pins are plugged into a tester head to make contact with the footprint pads.

Test for continuity is made by measuring typical current through the PCB conductor paths. Resistance measurements are used to indicated isolation and path resistances.

Another testing function which is performed before the actual assembly process is solderability testing. Both the PCB and surface mount components should be tested for solderability before they are assembled. This is a very important step since it is practically impossible to inspect the connections after they are soldered. Some automatic solderability testers are already available on the market.

Post assembly functional testing of the PCB will probably remain unchanged regardless of the current method used. The method used for DIP boards should be adequate for surface-mount boards. Depending on the testing equipment used some special care will have to be taken.

If automatic test equipment is used, some adjustment will have to be made to account for faster switching cycle, or processing time. These times may be faster because of the greater density and shorter circuit paths provided by surface mount assembly.

If using a bed-of-nails tester, it will be necessary to design the bed-of-nails to coincide with the new PCB design for the surface-mount components. Also some effort should be made to reduce the number of contact points and the nail contact size so that a smaller pad can be used on the PCB.

**Workability**

- Most of Monolithic Memories' products are currently available in PLCCs and SOs
- MMI's PLCC and SO packages conform to JEDEC standards for surface mount devices.
- More efficient layouts are possible since metal interconnects do not have to work around holes in PCBs.
- A wide range of discrete and active devices are already available in surface mount packages and the number is rapidly increasing
- An infrastructure of surface mount vendors and services is currently available to assist in production or conversion to surface mount

**Repair Techniques**

When a faulty device has been identified, it is necessary to remove this component and replace it with a new one. One approach would be to reflow the whole board, remove the faulty component and insert a new one while the solder is still liquid. This procedure is not recommended because it endangers the reliability of the other solder joints. Techniques which affect only the faulty component are more reliable.

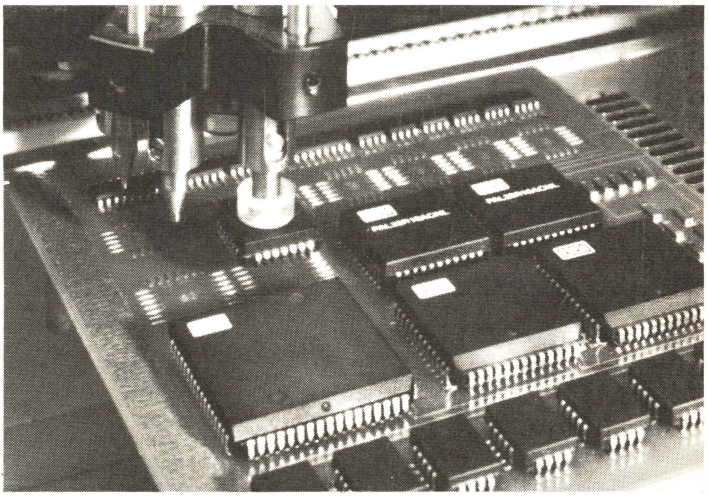
One way to solder or unsolder only one surface mount component is to use a heated collet. A component is inserted in the collet which comes in contact with the PCB. Heat is applied until the solder reflows, at which time the faulty device can be removed and a new one inserted.

Heated probes are another alternative for PLCCs. The tool consists of two probes attached in a scissor fashion and sized to fit the four sides of a chip carrier. The probes are heated like a conventional iron and can be used both to remove and install single surface mount components. Some sacrifice in board density may have to be made if the designer anticipates to use probes for repair, in order to allow sufficient space between the components to insert the probe without melting the pads of two devices.

One technique that works well for closed-spaced components is the use of a hot air stream on the connections. Once the solder reflows, the component is removed with tweezers. In the same category is the use of superheated gases, such as nitrogen, nitrogen plus hydrogen, or argon to reflow the solder. The advantage of hot gas is the prevention of contamination and oxidation which may occur when hot air is used.

Another method for reflowing surface mount connections of closely spaced components is the use of heat from an infrared light which is focused on the component.

As the usage of surface mount components increases, manufacturers of repair tools and equipment are responding with a greater number of options from which the potential user may choose. The same methods can be used for PLCCs or SO packages. The SO packages are somewhat easier because the leads are more accessible and on two sides only.



# The Plastic Leaded Chip Carrier

The Plastic Leaded Chip Carrier (PLCC) is the square, JEDEC-standard plastic package. Pins are located on all four sides of the package and spaced at 50 mils, significantly reducing the component size over the 100-mil spaced DIP. Leads are bent down and under the molded body, forming a "J-Hook" shape. This protects them from damage and entanglement during handling and shipment, making them excellent for use in automated assembly. Parts are currently available in 20-, 28-, 44-, 68-, and 84-pin packages. PLCCs are manufactured with the same process and materials as the conventional DIP for high reliability and low cost. All MMI's newly released 28-pin devices have adopted the JEDEC approved pinout with center no-connect pins where applicable.

## Advantages

- Most suitable package for automated assembly
- Easy to handle
- High copper content leadframe for better heat dissipation
- Highest space efficiency
- High reliability
- J-Hook leads absorb thermal and mechanical stress
- Low cost
- JEDEC standard package
- Improved electrical performance over DIPs
- Mountable on both sides of circuit board
- Solder-coated leads for easy, reliable soldering

## PLCC vs. DIP

| NUMBER OF PINS | PACKAGE DIMENSIONS L x W (inches) | | AREA (square inches) | | AREA RATIO (PLCC/DIP) |
|---|---|---|---|---|---|
| | PLCC | DIP | PLCC | DIP | |
| 20 | 0.35 x 0.35 | 1.0 x 0.25 | 0.12 | 0.26 | 0.46 |
| 24 (SKINNYDIP®) | — | 1.2 x 0.25 | — | 0.30 | 0.66 |
| 24 (Wide) | — | 1.2 x 0.55 | — | 0.66 | 0.30 |
| 28 | 0.45 x 0.45 | 1.5 x 0.55 | 0.20 | 0.83 | 0.24 |
| 40 | — | 2.1 x 0.55 | — | 1.16 | — |
| 44 | 0.65 x 0.65 | — | 0.42 | — | 0.36 |

# Thermal Characteristics

The thermal characteristics of PLCCs resemble those of corresponding DIP packages. Monolithic Memories' plastic leaded chip carriers employ a high-copper content leadframe to aid in heat removal.

## Thermal Impedance Calculation

$\theta_{ja} = \theta_{jc} + \theta_{ca}$,

where

$\theta_{ja}$ = Thermal impedance from junction to ambient, °C/W
$\theta_{jc}$ = Thermal impedance from junction to case, °C/W
$\theta_{ca}$ = Thermal impedance from case to ambient, °C/W.

$T_j = Ta + (P \times \theta_{ja})$

$T_j$ = Junction temperature, °C
$T_a$ = Ambient temperature, °C
P = Power dissipation, Watts.

$P = I_{CC} \times V_{CC}$

$I_{CC}$ = Device current, Amps
$V_{CC}$ = Device supply voltage, Volts.

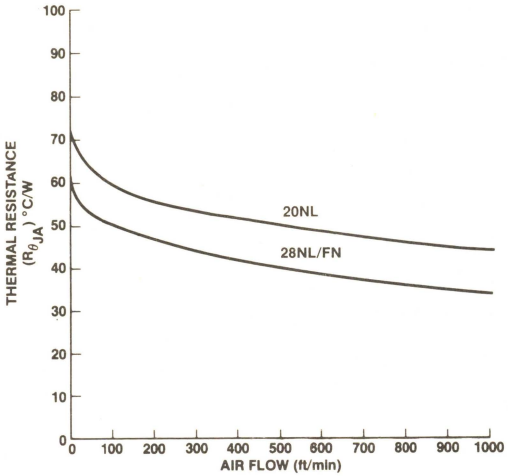
## Sample Calculation

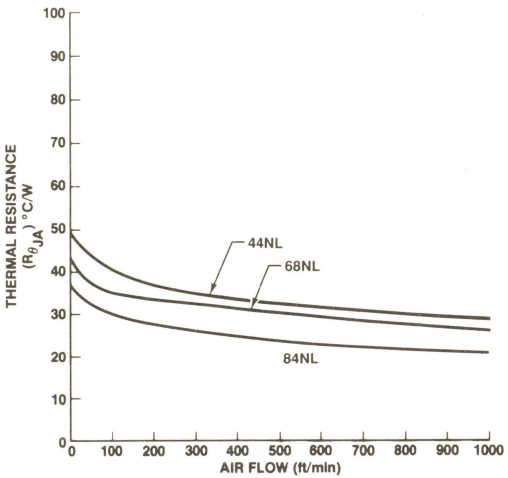
28NL package
Air Flowrate = 0 ft/s
$T_a = 75°C$
$V_{CC} = 5.0$ V
$I_{CC} = 180$ mA

From figure, $\theta_{ja} = 61.0°C/W$
Calculate $P = 5.0 \times 0.180 = 0.9$ W
$T_j = 75 + (0.9 \times 61.0)$
$= 129.9°C$

## Thermal Expansion

The PLCC package body is made of moisture-resistant, thermally-conductive epoxy resin. Its thermal coefficient of expansion (TCE) matches that of most epoxy board materials. PLCCs can also be used on substrates with a different TCE, because the J-Hook leads absorb mismatch stress.




| PACKAGE | DIE SIZE $(mils)^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20NL | 11,250 | 14 |
| 28NL/FN | 22,500 | 13 |
| 44NL | 22,500 | 11 |
| 68NL | 50,625 | 8 |
| 84NL | 50,625 | 6 |

*These are typical values for the given die size.

# The Small Outline Gull-wing Package (SO)

The Small Outline package has dual in-line leads spaced 50 mils on center, compared to 100 mils on the DIP. This gives the SO package the appearance of the DIP with the compactness of the PLCC. The leads of the SO Gull-wing package start out like the DIP's but are bent out to rest flat on a board. Although it increases the width of the overall package, it makes soldering easier and provides a reliable solder joint. Parts are currently available in 16-, 20-, 24-, and 28-pin packages. The SO Gull-wing package follows proposed JEDEC standards for surface mount packages.

## Advantages

- Suitable for automated assembly
- Easy to handle
- Increased board density
- Reduced inventory storage floor space
- Reduced manufacturing time
- Easily inspectable gull-wing leads can be probed by test leads
- Flexible leads can absorb thermal expansion mismatches between the package and the board
- Low cost
- Mountable on both sides of circuit board

## SO vs DIP

| NUMBER OF PINS | PACKAGE DIMENSIONS L x W (inches) | | AREA (square inches) | | AREA RATIO (SO/DIP) |
|---|---|---|---|---|---|
| | SO | DIP | SO | DIP | |
| 16 | 0.40 x 0.30 | 0.75 x 0.25 | 0.12 | 0.19 | 0.63 |
| 20 | 0.50 x 0.30 | 1.00 x 0.25 | 0.15 | 0.26 | 0.58 |
| 24 | 0.60 x 0.30 | 1.20 x 0.25 (SKINNYDIP) | 0.18 | 0.30 | 0.59 |
| | | 1.20 x 0.55 (Wide) | | 0.66 | 0.27 |
| 28 | 0.70 x 0.30 | 1.50 x 0.55 | 0.21 | 0.83 | 0.25 |

Table 1. SO vs. DIP Package Dimensions

# Thermal Characteristics

Thermal characteristics of surface mount devices were calculated by mounting devices in direct contact with a double-sided fiberglass-epoxy composite PBC.

For measurement of $R_{\theta ja}$ all packages were immersed in a constant temperature fluorinert bath.

## Thermal Impedance Calculation

$\theta_{ja} = \theta_{jc} + \theta_{ca}$,

where

$\theta_{ja}$ = Thermal impedance from junction to ambient, °C/W
$\theta_{jc}$ = Thermal impedance from junction to case, °C/W
$\theta_{ca}$ = Thermal impedance from case to ambient, °C/W.

$T_j = Ta + (P \times \theta_{ja})$

$T_j$ = Junction temperature, °C
$T_a$ = Ambient temperature, °C
P = Power dissipation, Watts.

$P = I_{CC} \times V_{CC}$

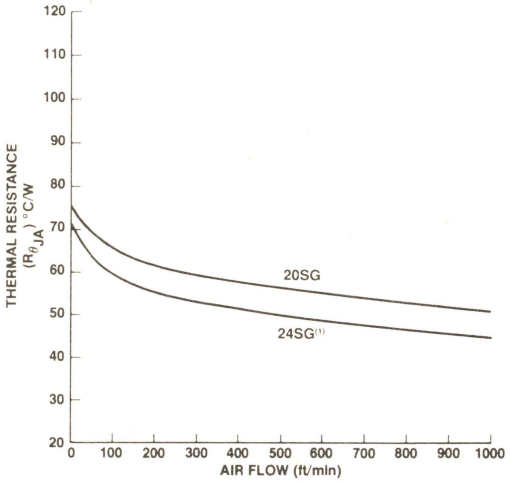
$I_{CC}$ = Device current, Amps
$V_{CC}$ = Device supply voltage, Volts.

## Sample Calculation

24SG package
Air Flowrate = 0 ft/s
$T_a = 75°C$
$V_{CC} = 5.0$ V
$I_{CC} = 180$ mA

From figure, $\theta_{ja} = 73.0°C/W$
Calculate $P = 5.0 \times 0.180 = 0.9$ W

$T_j = 75 + (0.9 \times 73.0)$
$= 140.7°C$




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20SG | 5,625 | 16 |
| 24SG(1) | 11,250 | 13 |
| 24SG(2) | 22,500 | 10 |

* These are typical values for the given die size.

3

## Surface Mount Packing

Monolithic Memories' surface mount packages are available in standard tubes with varying parts counts for different size packages.

PLCC and SO devices are also available in tape-and-reel format. Components are encapsulated in a plastic ribbon with sprocket holes, similar to a movie film. Tape-and-reel packaging offers the best control over the component feeding process. It also greatly increases the time between reloading as compared to tube feed. Reels contain therefore many more devices and don't need to be reloaded as often as tubes.

These advantages of tape-and-reel packing will become even more important as board assembly moves toward complete automation.

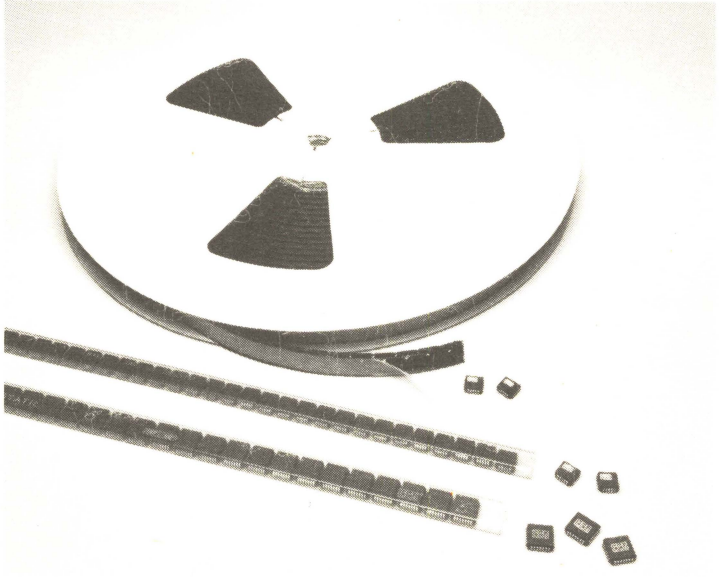
Monolithic Memories' commitment to component quality, through our Product Assurance Program, eliminates the need for incoming component screening. Therefore customers can take advantage of tape-and-reel packing without sacrificing confidence in product quality.

### Tube Format

| PACKAGE SIZE | PLCC PARTS PER TUBE | SO PARTS PER TUBE |
|---|---|---|
| 16 SG | | 50 |
| 20 SG/NL | 50 | 40 |
| 24 SG | | 30 |
| 28 SG/NL/FN | 40 | 25 |
| 44 NL | 35 | |
| 68 NL | 20 | |
| 84 NL | 15 | |

### Tape-and-Reel Format

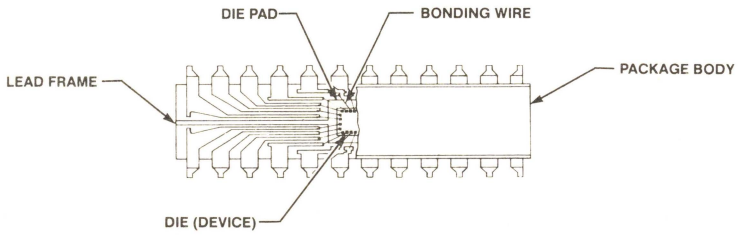
| PACKAGE SIZE | NUMBER OF PARTS PER REEL | TAPE WIDTH (MM) |
|---|---|---|
| 16 SG | 1,400 | 16 |
| 20 SG | 1,400 | 24 |
| 20 NL | 1,200 | 16 |
| 24 SG | 1,400 | 24 |
| 28 SG | 1,400 | 24 |
| 28 NL/FN | 950 | 24 |
| 44 NL | 640 | 32 |



# PAL Device Package Outlines

## Package Drawing

## Molded DIP




**LEAD FRAME**

Copper Alloy 194.
Copper Alloy Tamac 5.

**BONDING WIRE**

1.0 Mil Gold Wire.
1.25 Mil Gold Wire.
1.30 Mil Gold Wire.

**PACKAGE BODY**

Thermoset Plastic.

**LEAD FINISH**

Solder Dip.

**DIE PAD**

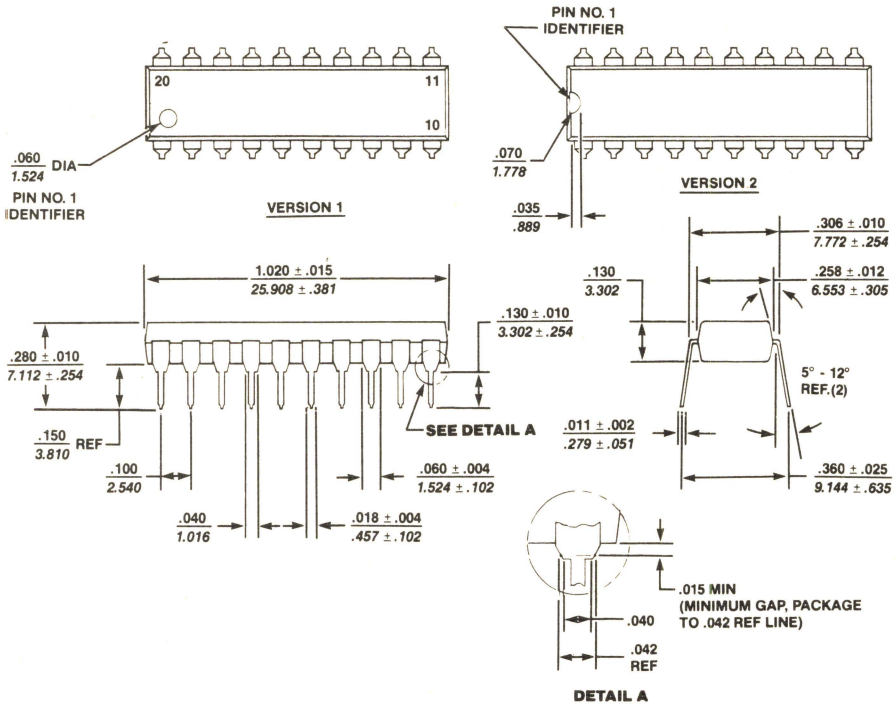
Spot Silver Plating
(150 Micro-Inches)

**DIE BOND**

Silver Filled Epoxy.

## Package Drawings

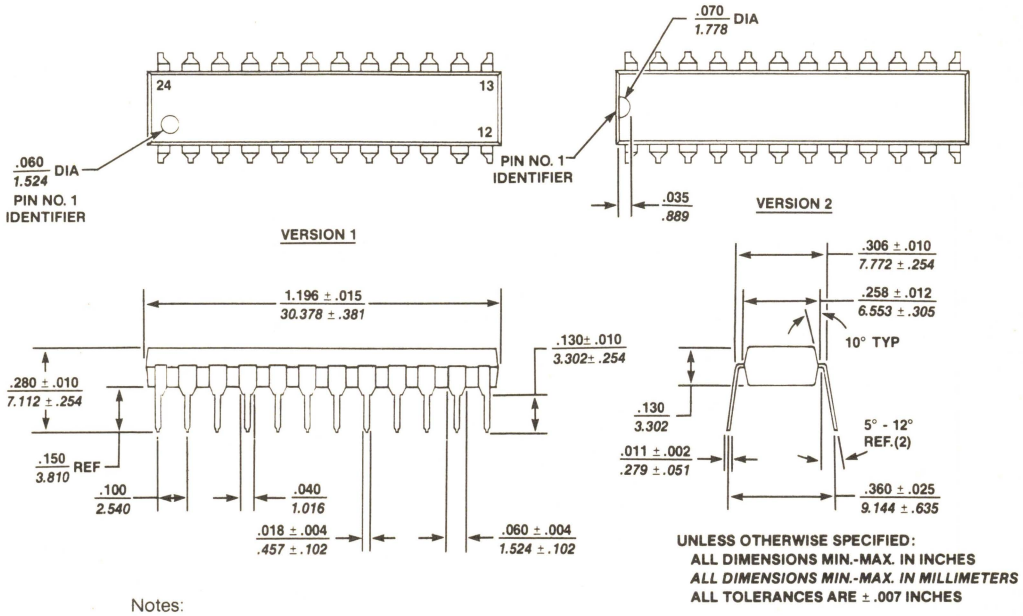
20N Molded DIP
(1/4"x1")




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

**24NS Molded SKINNYDIP**
**(1/4"x1 3/16")**

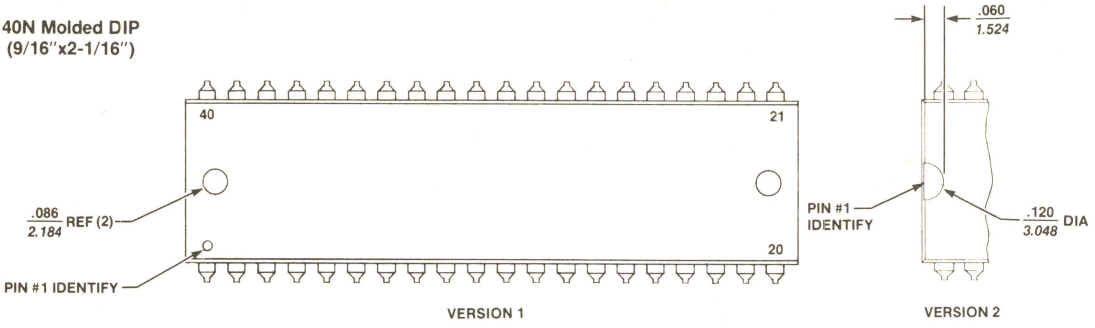


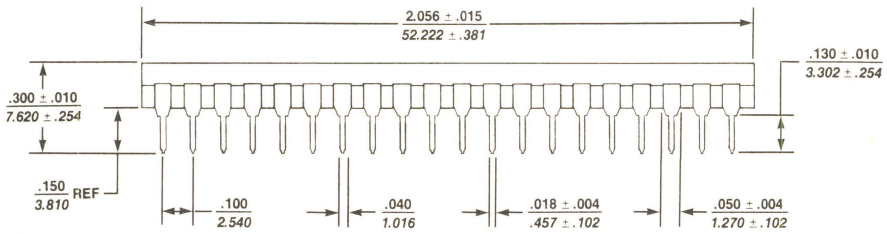

Notes:

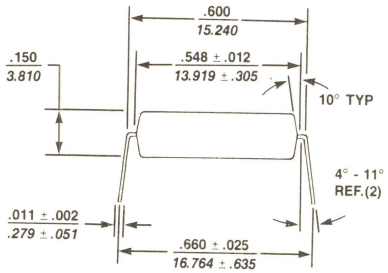
1. Lead material tolerances are for tin plate finish only. Solder dip finish adds 2-10 mils thickness to all lead tip dimensions.
2. Both version 1 and version 2 configurations are manufactured interchangeably.
3. Ejector pin marks on version 1 are optional.

## Package Drawing

40N Molded DIP
(9/16"x2-1/16")



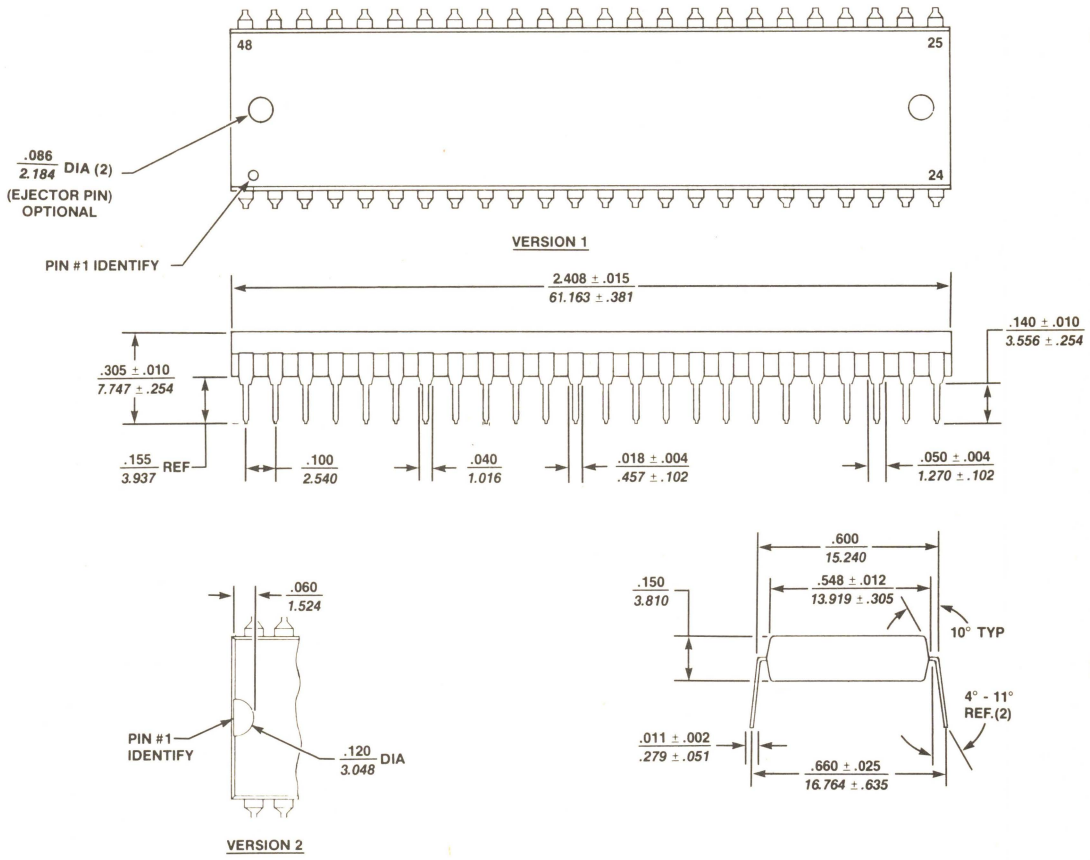




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

**48N Molded DIP**
**(9/16"x2 13/32")**



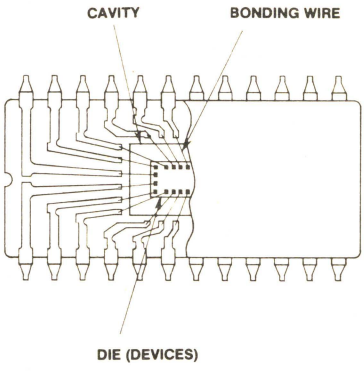

**UNLESS OTHERWISE SPECIFIED:**
**ALL DIMENSIONS MIN.-MAX. IN INCHES**
***ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS***
**ALL TOLERANCES ARE ± .007 INCHES**

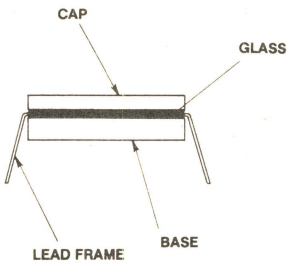
Notes:

1. Lead material tolerances are for tin plate finish only. Solder dip finish adds 2-10 mils thickness to all lead tip dimensions.
2. Both version 1 and version 2 configurations are manufactured interchangeably.
3. Ejector pin marks on version 1 are optional.

## Package Drawing

# Ceramic DIP




**LEAD FRAME**
Alloy 42

**BONDING WIRE**
1.25 Mil Aluminum

**CAP AND BASE**
Pressed Alumina

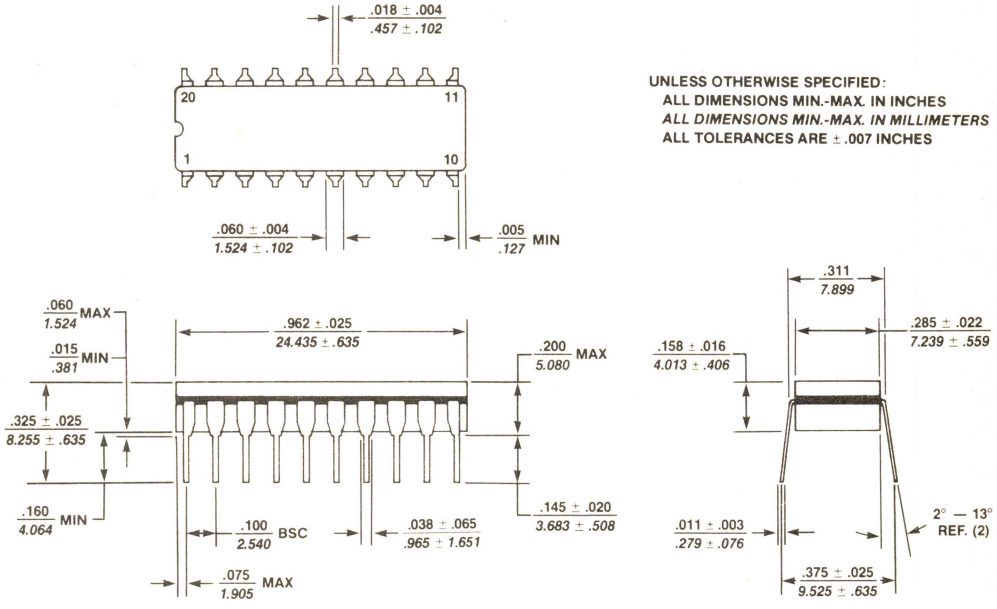
**GLASS**
Vitreous
Solder Glass

**CAVITY**
Gold Over Alumina
For Eutectic Die Attach

**LEAD FINISHES**
Solder DIP Over
Matte Tin Plate

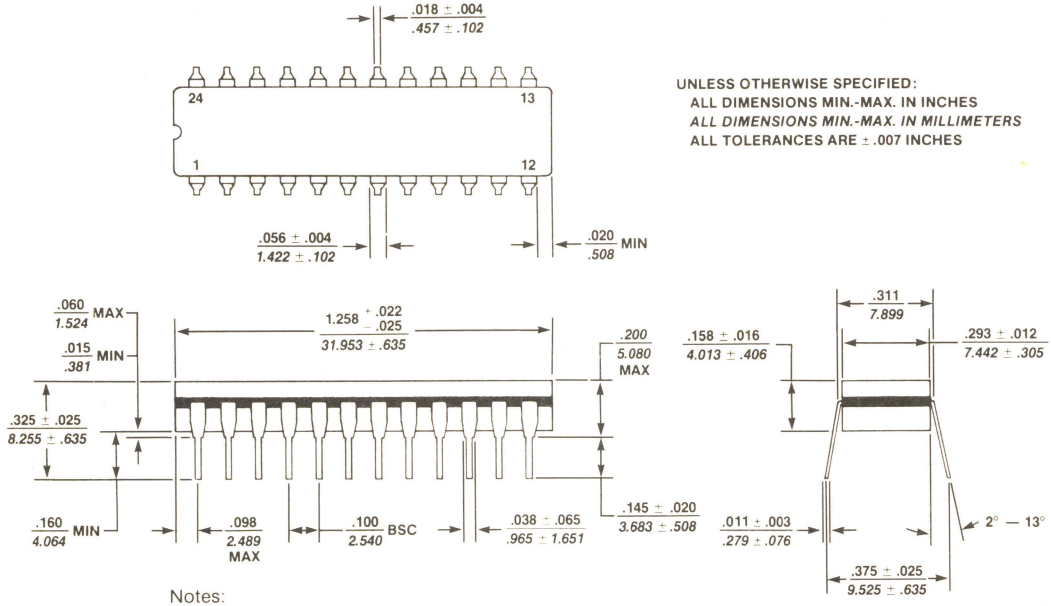
## Package Drawing

20J Ceramic DIP
Mil-M-38510,
Appendix C, D-8



## Package Drawing

24JS Ceramic SKINNYDIP
Mil-M-38510,
Appendix C, D-9

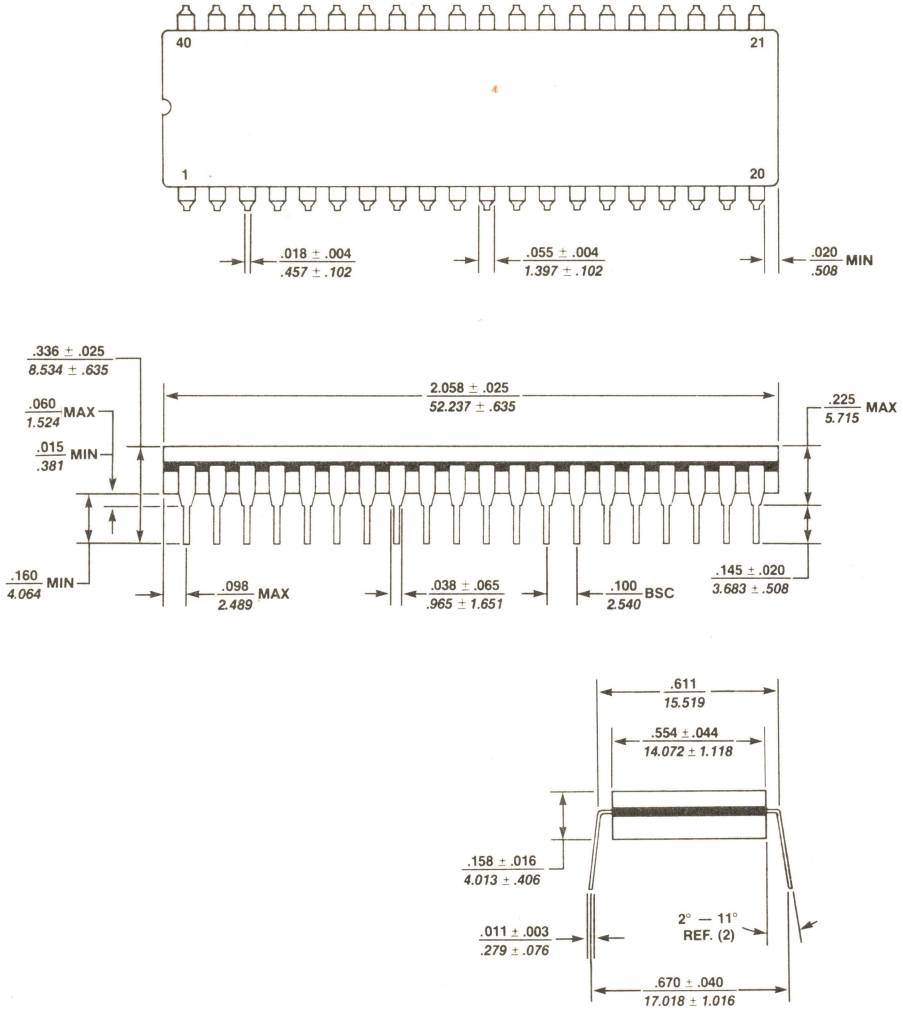



Notes:

1. Specified body dimensions allow for differences between MSI and LSI packages.
2. Lead material tolerances are for tin plate finish only. Solder dip finish adds 2-10 mils thickness to all lead tip dimensions.

## Package Drawing

40J Ceramic DIP
Mil-M-38510,
Appendix C, D-5



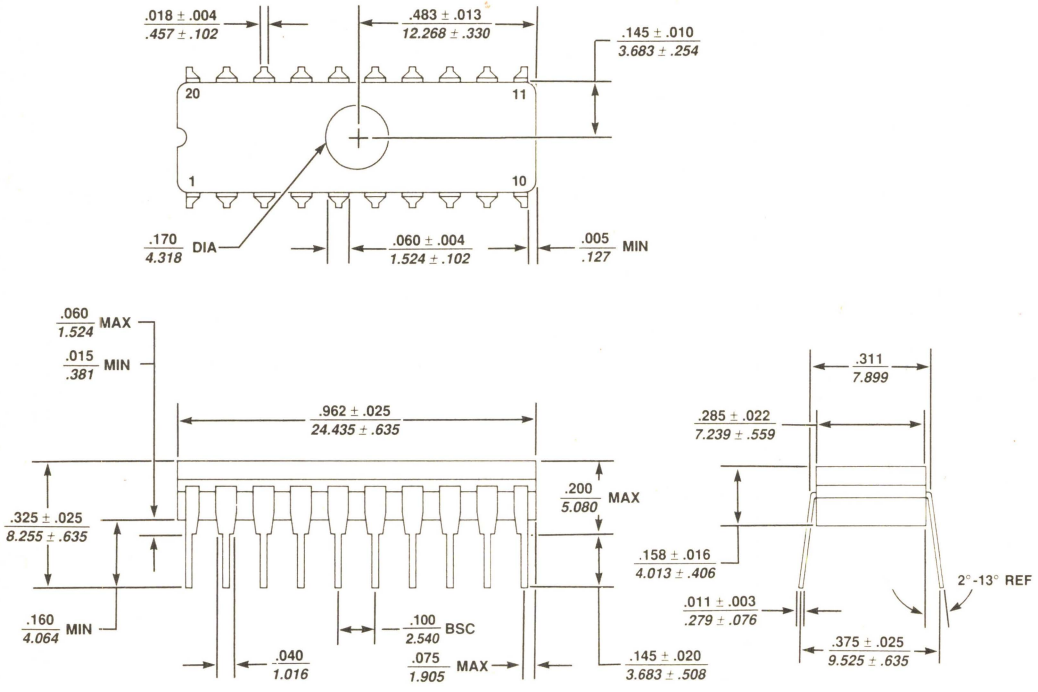

**UNLESS OTHERWISE SPECIFIED:**
**ALL DIMENSIONS MIN.-MAX. IN INCHES**
***ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS***
**ALL TOLERANCES ARE ± .007 INCHES**

Notes:

1. Specified body dimensions allow for differences between MSI and LSI packages.
2. Lead material tolerances are for tin plate finish only. Solder dip finish adds 2-10 mils thickness to all lead tip dimensions.

## Package Drawing

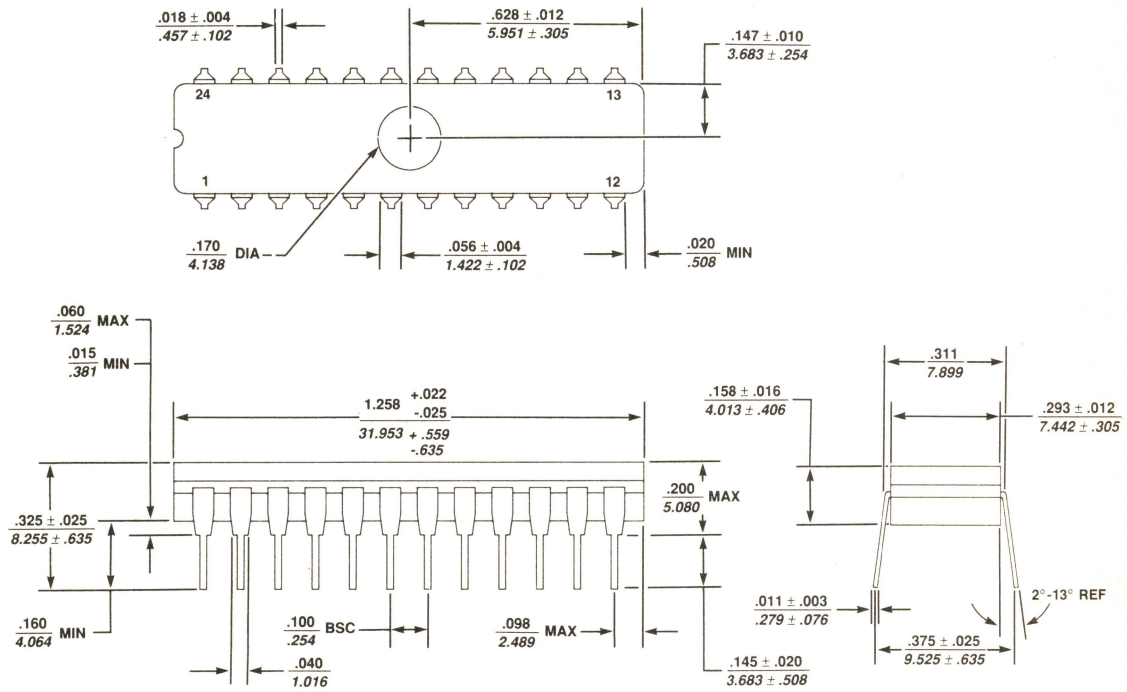
20 Q Window CERDIP




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

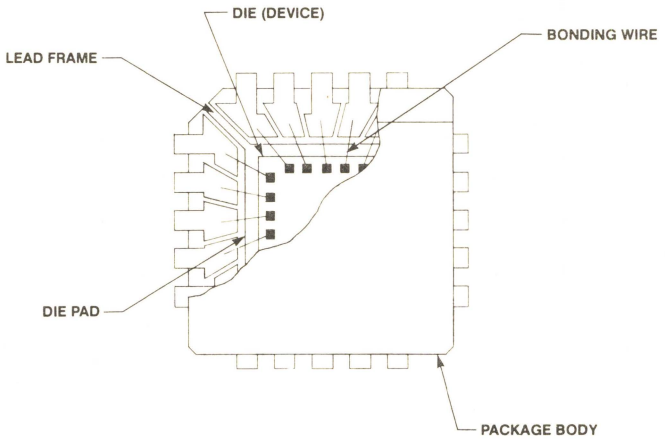
24-QS Window CERDIP
(5/16″ x 1 1/4″)




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

# Plastic Leaded Chip Carrier




**LEAD FRAME**

Copper Alloy 195.
Copper Alloy Tamac 5.

**BONDING WIRE**

1.25 Mil Gold Wire

**PACKAGE BODY**

Thermoset Plastic.

**LEAD FINISH**

Tin Plating.
Solder Dip.

**DIE PAD**

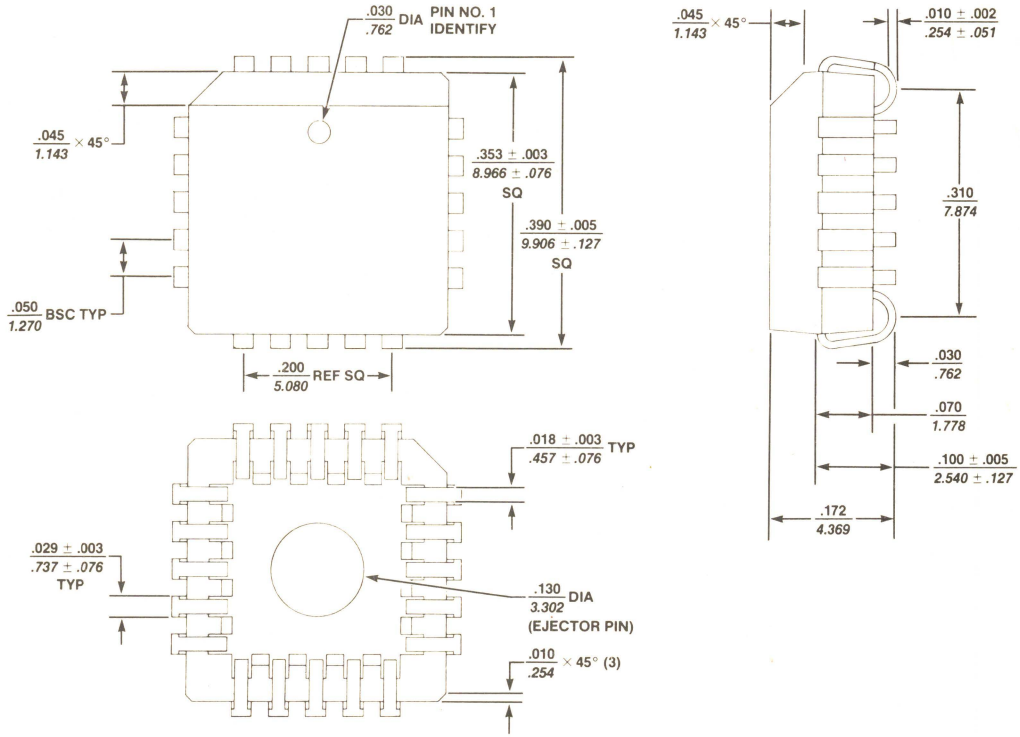
Spot Silver Plating
(150 Microinches).

**DIE BOND**

Silver Filled Epoxy.

## Package Drawing

20NL Plastic Leaded Chip Carrier
(.351″ x .351″)

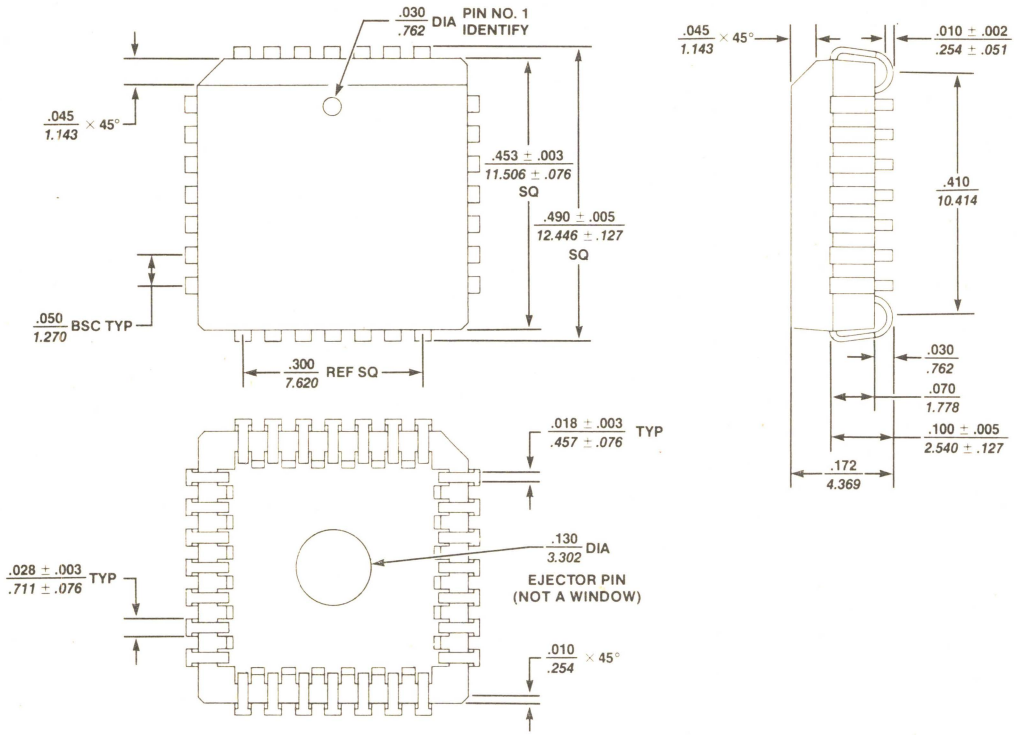



UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

3

## Package Drawing

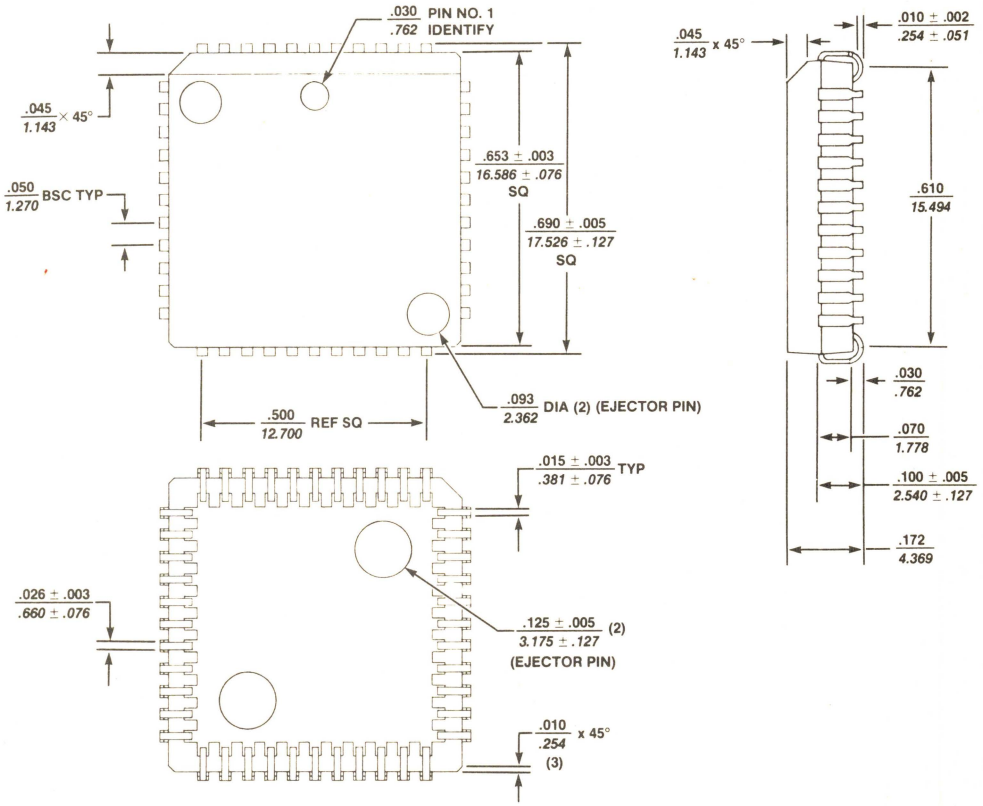
28NL/FN Plastic Leaded Chip Carrier
(.451" x .451")




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

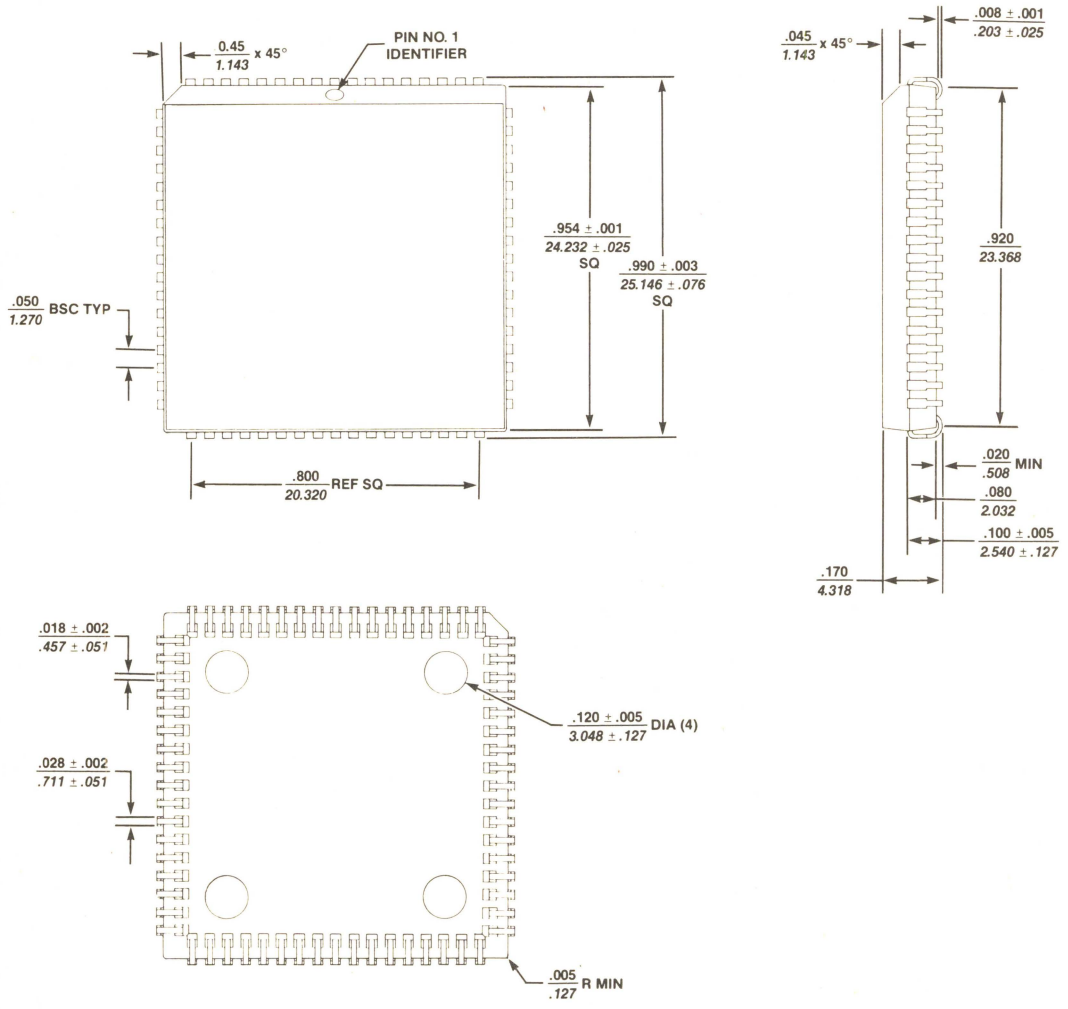
44NL Plastic Leaded Chip Carrier
(.650" x .650")




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

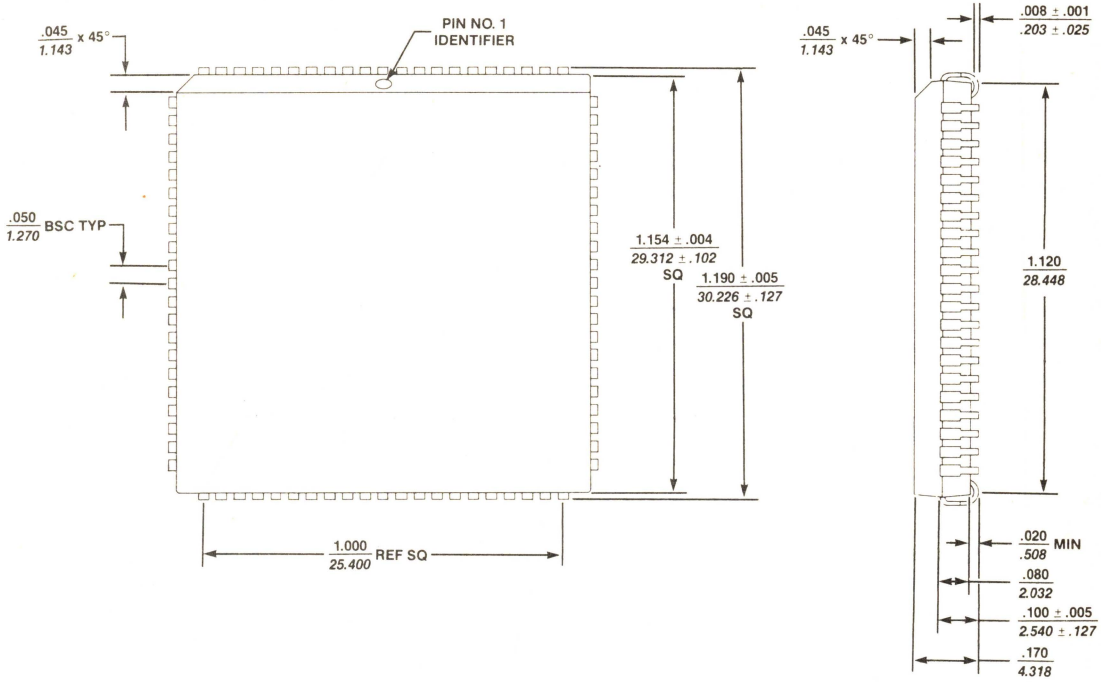
68NL Molded Chip Carrier
(.950"x.950")

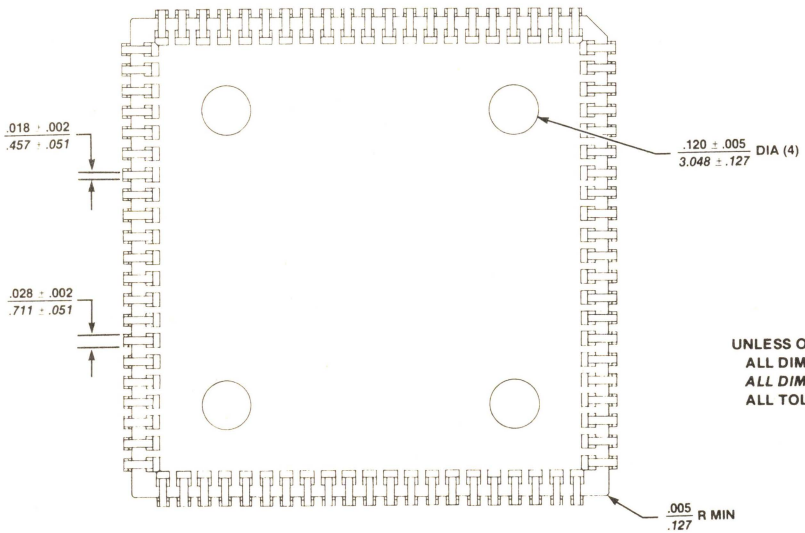



UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

84NL Plastic Leaded Chip Carrier
(1.154" x 1.154")

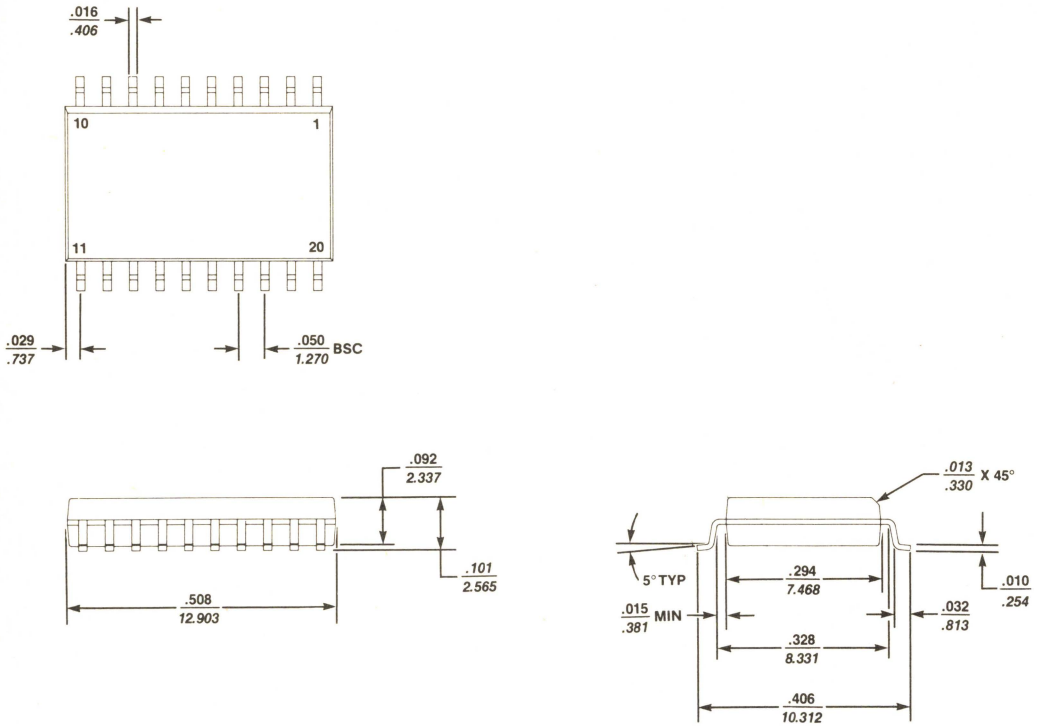



UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

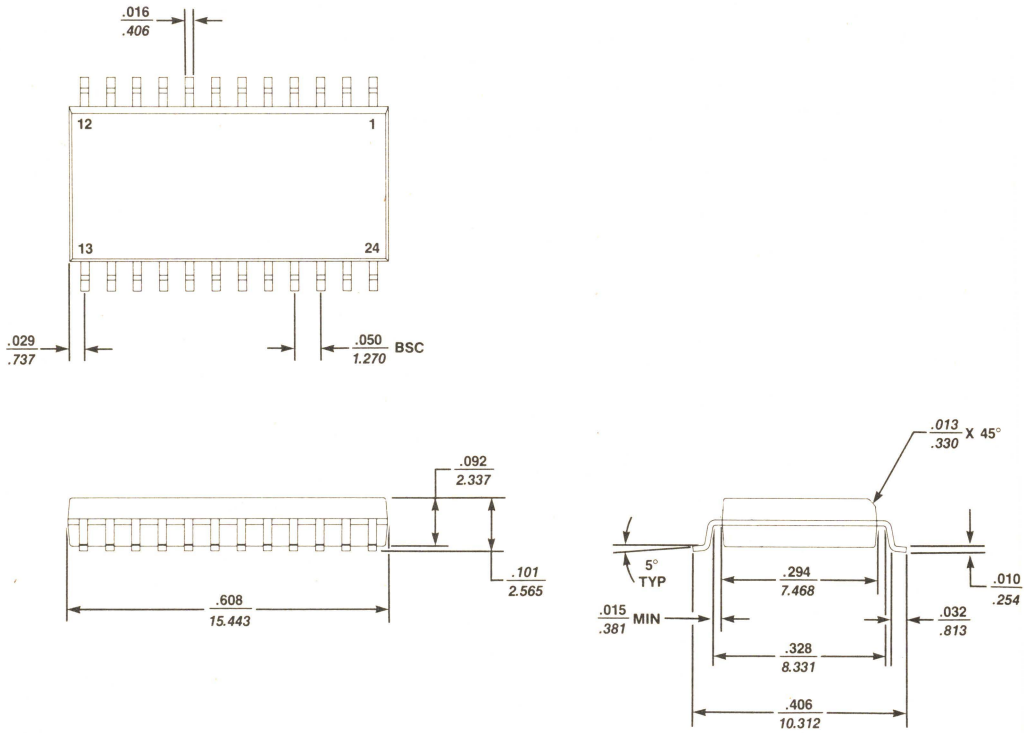
20SG Small Outline Package




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

24SG Small Outline Package

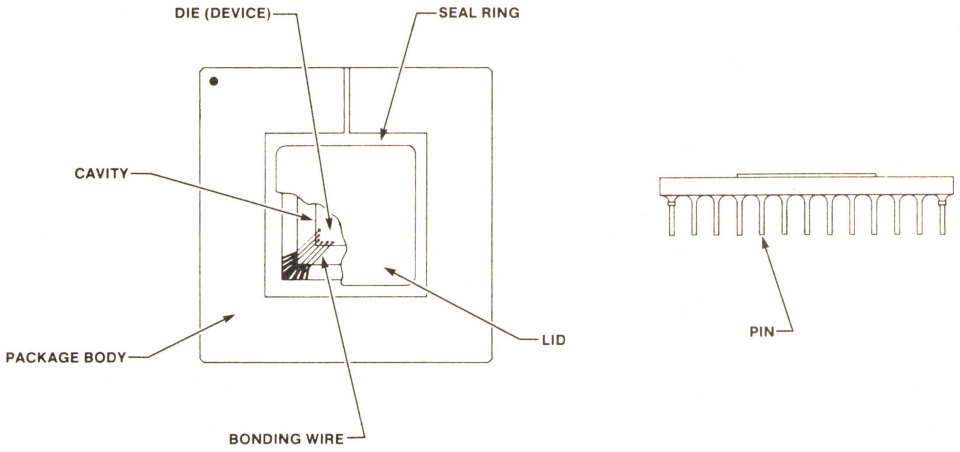



UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

3

## Package Drawing

# Pin Grid Array




**PACKAGE BODY**

Alumina
(Standard Dark)

**BONDING WIRE**

1.25 Mil Aluminum

**LID**

Gold Plated Kovar With
Nickel Underplating

**CAVITY/SEAL RING**

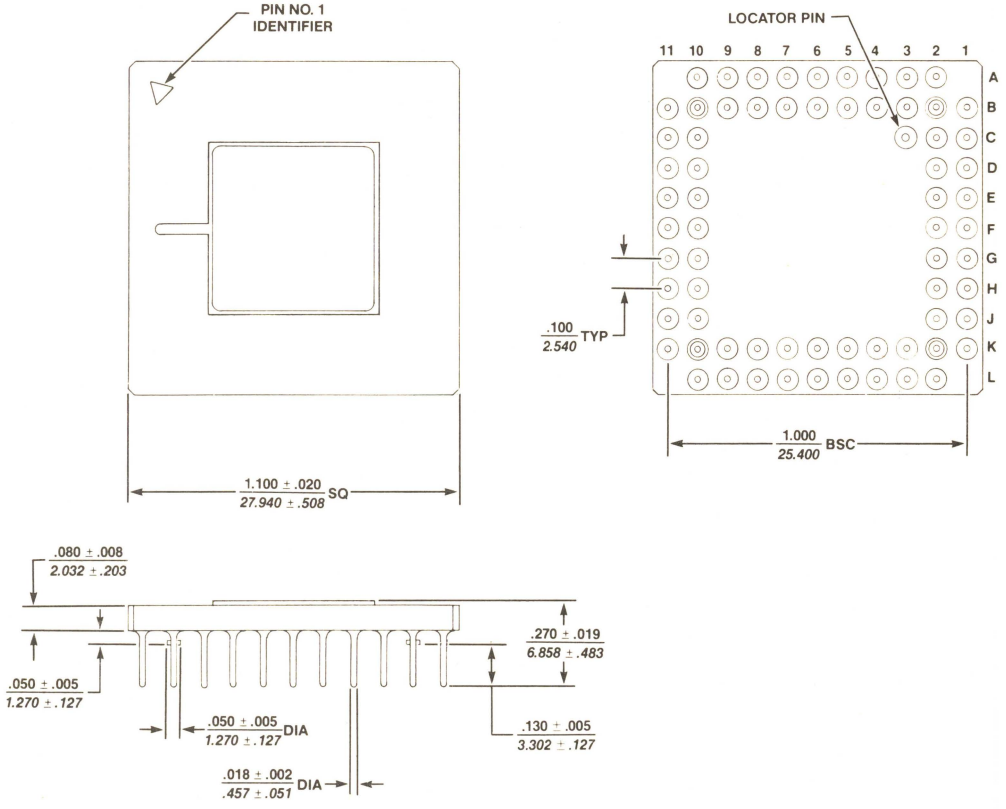
Gold Over Tungsten

**PIN MATERIAL**

Gold Plated Kovar

## Package Drawing

68P Ceramic Pin Grid Array

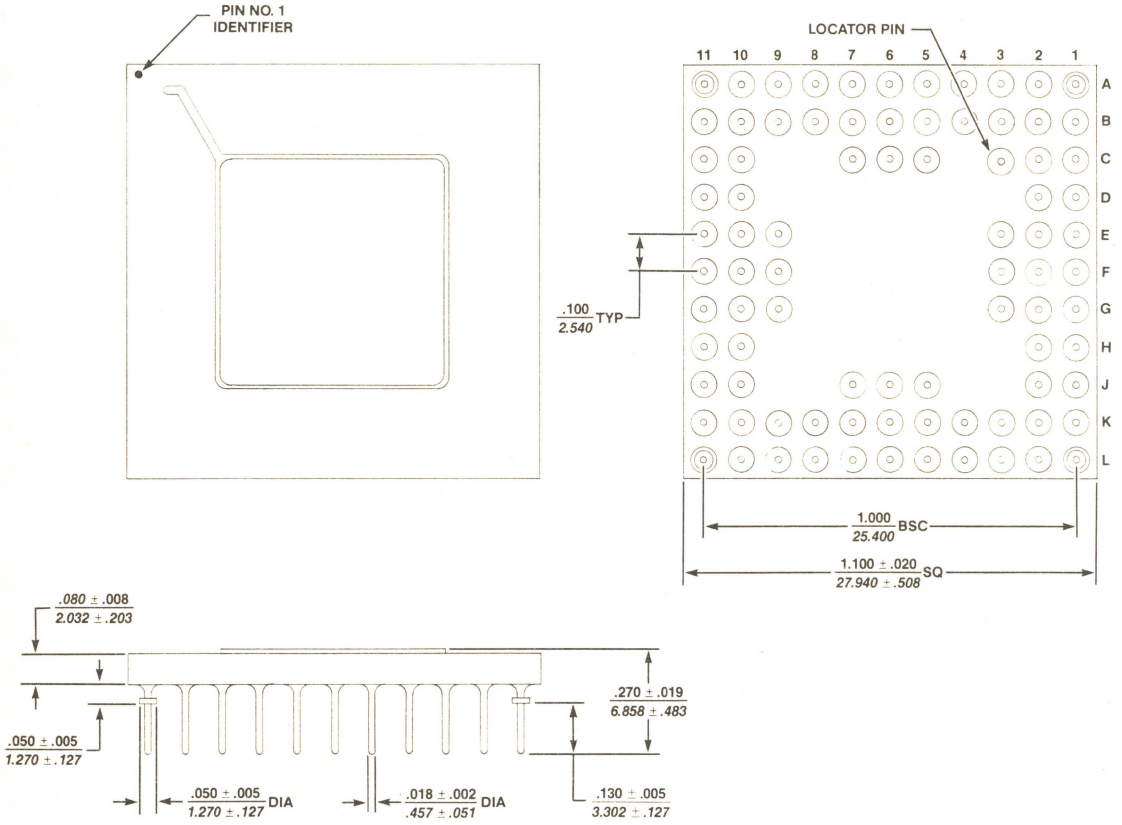



UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

3

## Package Drawings

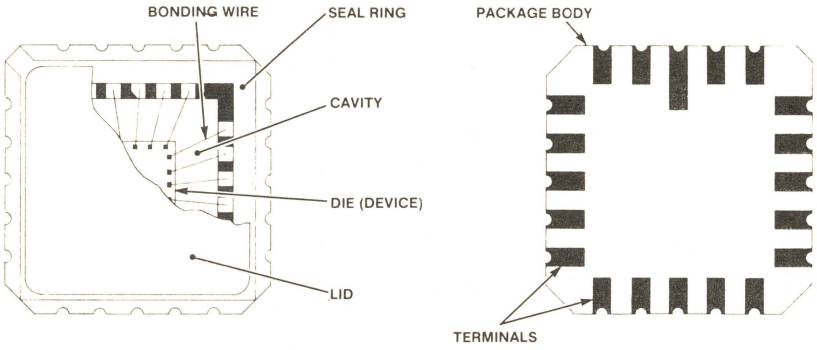
84P Ceramic Pin Grid Array




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

# Leadless Chip Carrier




**PACKAGE BODY**

Alumina
(Standard Dark)

**BONDING WIRE**

1.25 Mil Aluminum

**LID**

Gold Plated Kovar With
Nickel Underplating

**CAVITY/SEAL RING**

Gold Over Nickel
Over Tungsten

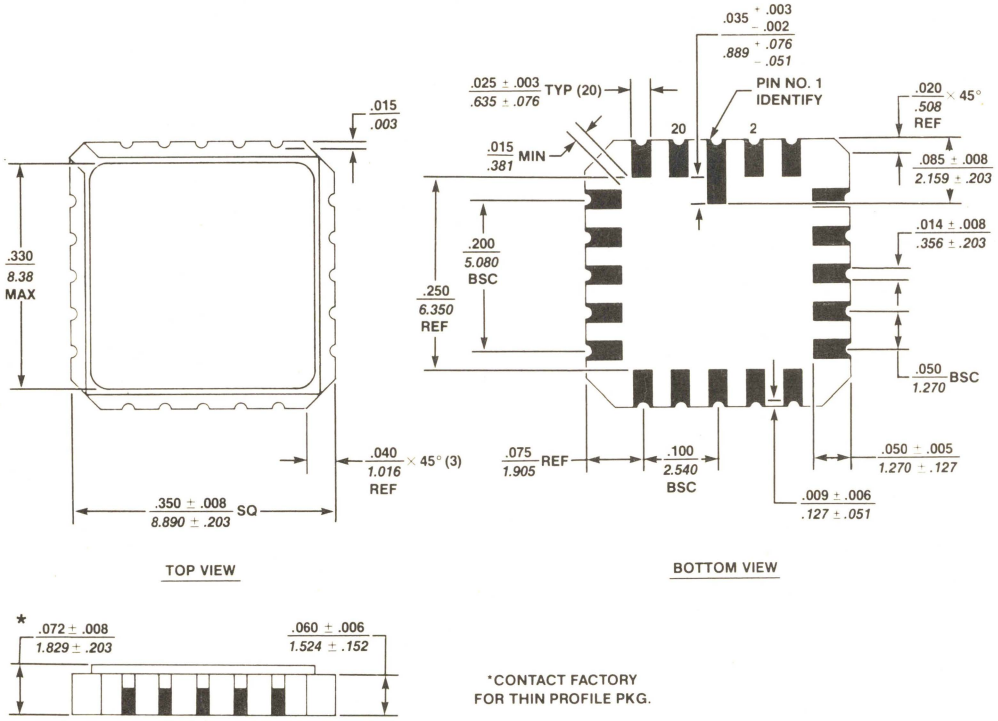
**TERMINALS**

Gold Plating Over Tungsten

3

## Package Drawing

20L Leadless Chip Carrier
Mil-M-38510,
Appendix C, C-2

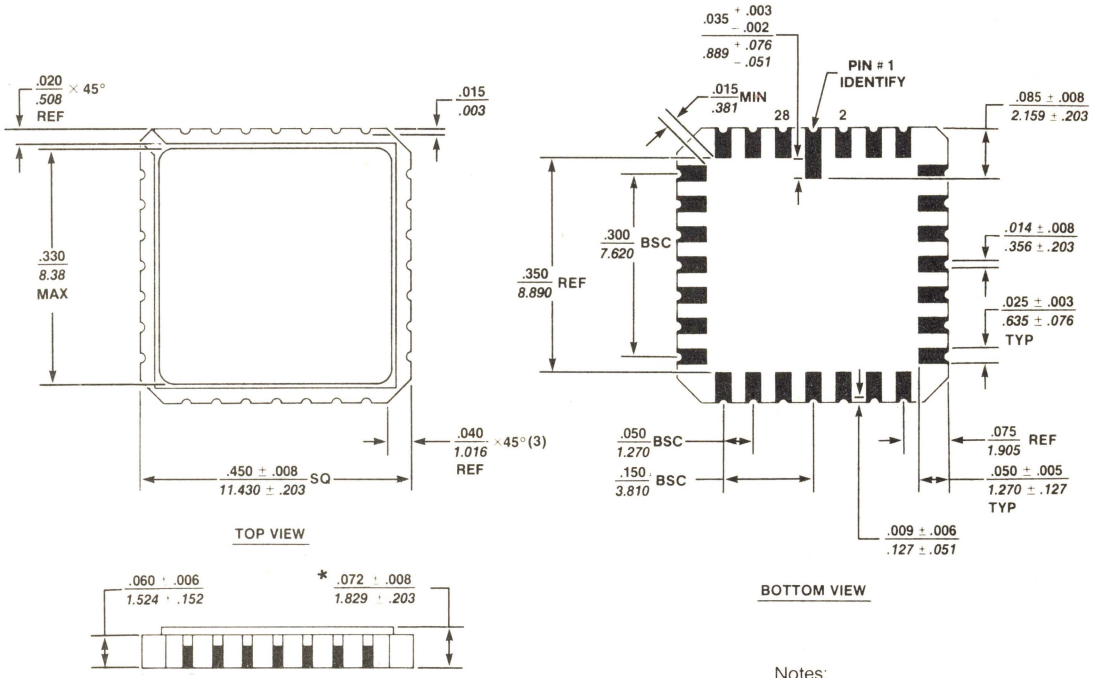



*CONTACT FACTORY
FOR THIN PROFILE PKG.

UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

28L Leadless Chip Carrier
Mil-M-38510,
Appendix C, C-4




*CONTACT FACTORY FOR THIN PROFILE PKG.

Notes:

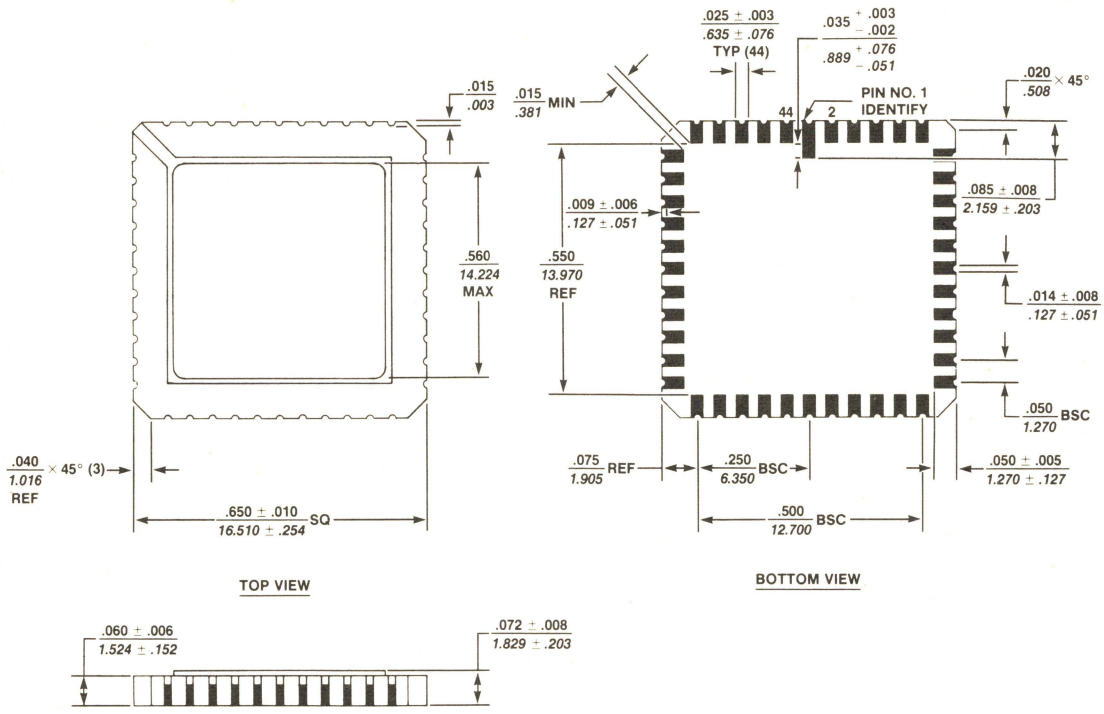
1. Solder fillets on lid edges not shown.

UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

3

## Package Drawing

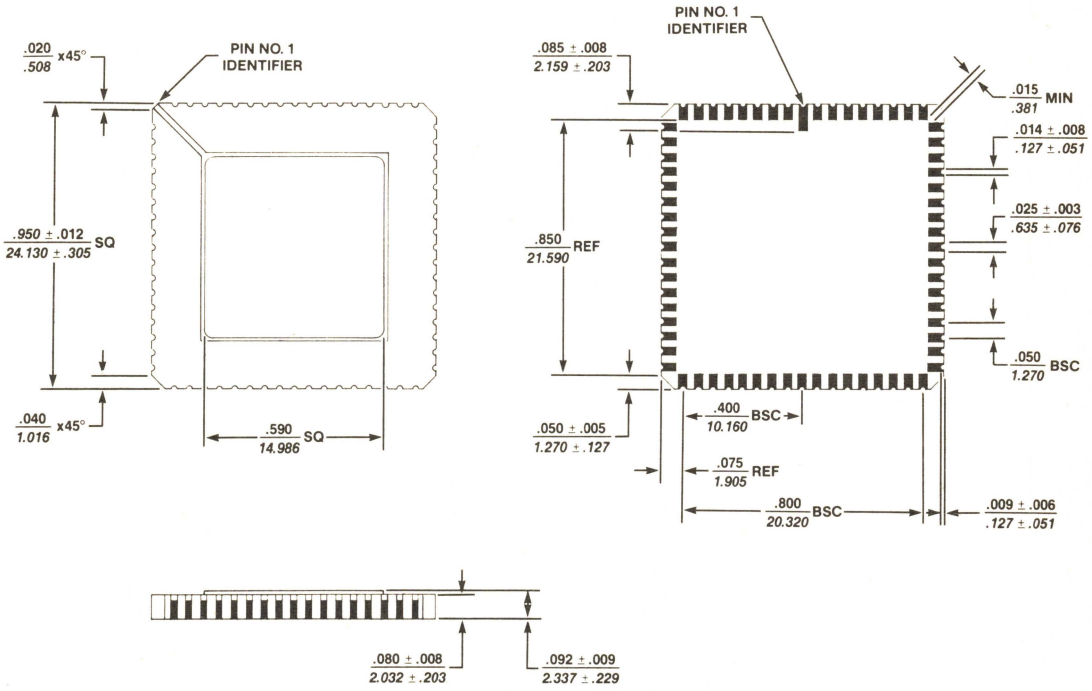
44L Leadless Chip Carrier
Mil-M-38510,
Appendix C, C-5




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

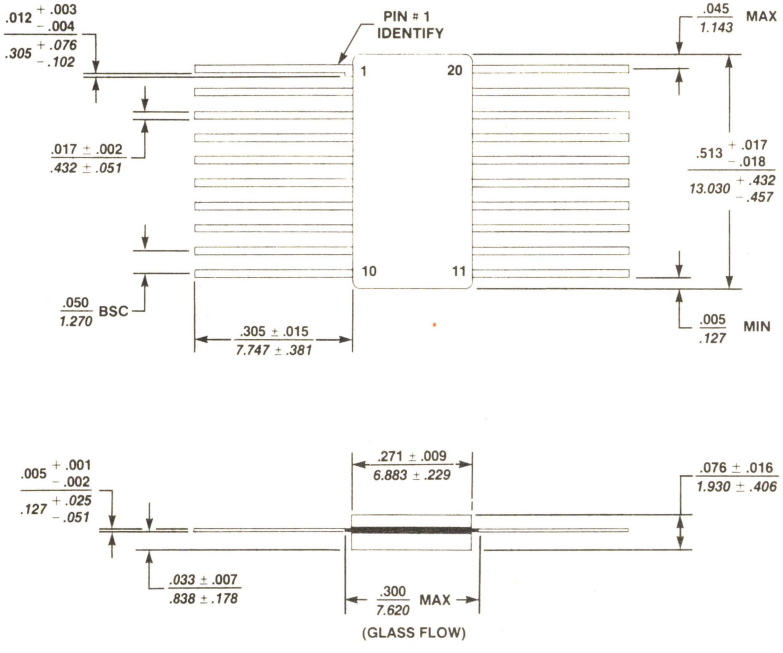
68L Leadless Chip Carrier
MIL-M-38510
Appendix C, C-7




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

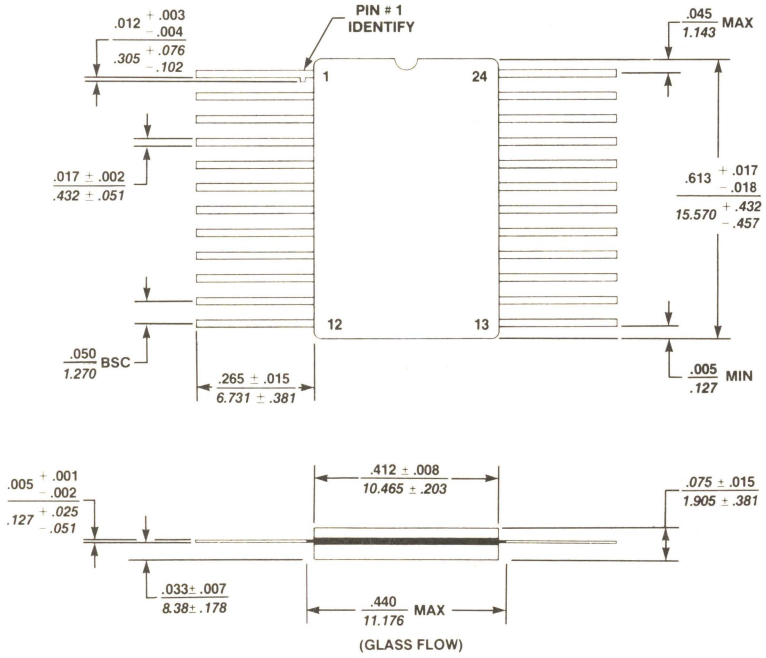
20W Cerpack
Mil-M-38510,
Appendix C, F-9




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ± .007 INCHES

## Package Drawing

24W Cerpack
Mil-M-38510,
Appendix C, F-6




UNLESS OTHERWISE SPECIFIED:
ALL DIMENSIONS MIN.-MAX. IN INCHES
*ALL DIMENSIONS MIN.-MAX. IN MILLIMETERS*
ALL TOLERANCES ARE ±.007 INCHES

3

# PAL Device Package Thermal Characteristics

## Introduction

Thermal resistance for a packaged integrated circuit determines the operating temperature and hence the performance and lifetime of the semiconductor device. For this reason, it is of interest to know the thermal resistance of the package configurations commonly in use and the effect of external factors such as air circulation and board-mounting conditions on the device temperature. To accomplish this end, measurement techniques and standards have been established providing certain conventions for data acquisition. Monolithic Memories has chosen to conform to these conventions in measurement and provides standard data for thermal resistance in the form of $R_{\theta JC}$ (resistance from junction to case) and $R_{\theta JA}$ (resistance from junction to ambient) as a function of air movement over the package/-board combination.

## Use of Monolithic Memories Data

In this publication, data is presented for a variety of packages and ambient conditions. In order to simplify the data presentation, graphs of $R_{\theta JA}$ vs. airflow are provided for packages in common use. These include socket-mounted pin grid arrays, dual-in-line p-dip, cerdip and side-brazed packages, board mounted cerpacks, flatpacks, leadless-chip carriers and plastic leaded chip carriers.

Resistance from junction to ambient ($R_{\theta JA}$) is a package geometry and die size related function. The user need only look up the package type and die size for the air-flow used. Since the $R_{\theta JC}$ is largely dependent on the package type and die size, a table has been constructed for easy use.

## Notes on the Tabulated Data

1. All side-brazed, cerdip-sealed, molded dual-in-line and pin grid array packages were mounted in zero insertion force sockets with 40 mils air gap and transverse to the airstream.
2. All cerpacks, flatpacks, LCC, PLCC and SOIC packages were board mounted in direct contact with a double-sided fiberglass-epoxy composite printed circuit board.
3. For measurement of $R_{\theta JC}$, all packages were immersed in a constant temperature fluorinert bath. The thermocouple was mounted directly to the bottom of the package.

10261A
JANUARY 1988

# Thermal Resistance Measurement Procedure

## Definition

Thermal resistance of a semiconductor device is a measure of the ability of its mechanical structure (package) to provide for heat removal from the semiconductor element. It is defined as the rise in the junction temperature against some reference point per unit power of dissipation or it may be described by the formula:

$$R_{\theta JR} = \frac{T_J - T_R}{P}$$

$R_{\theta JR}$ = Thermal resistance, junction to reference point, in °C/watt
$T_J$ = Junction temperature in °C
$T_R$ = Reference point temperature in °C
P = Power dissipation

## Thermal Measurement Technique

Thermal resistance is measured using the temperature sensitive parameter (TSP) method. This method takes advantage of the linear relation between temperature and voltage drop across a p-n junction to measure the average die temperature. Thermal resistance measurement can be done either using an actual device or with thermal test chips. For the purpose of this study, thermal test chips are used.

Each test chip consists of sensing elements and a heating element. Sensing elements are two sets of diode pairs. One diode pair is located at the center of each die and one pair is near a corner. The heating element is a polysilicon resistor which covers 95 percent of the die surface area. The resistor extends underneath the bond pads but not the sensing elements.

Initially, diodes are forward biased to a low level current source (50 $\mu$A) and the voltage drop is calibrated with respect to temperature. Then, the resistor is powered and the diode voltage drop is monitored until thermal equilibrium is reached. Steady state junction temperature is calculated from the calibration data.

For the $R_{\theta JA}$ measurement the device is put in a wind tunnel. The air speed is adjustable from 0 to 1000 feet/min. The use of a wind tunnel allows us to graph the $R_{\theta JA}$ vs. air flow velocity. Average junction to case thermal resistance ($R_{\theta JC}$) is measured by immersing the package in a constant temperature fluorinert bath and sensing steady state junction temperature with case temperature being measured at the bottom of the package.

## Summary

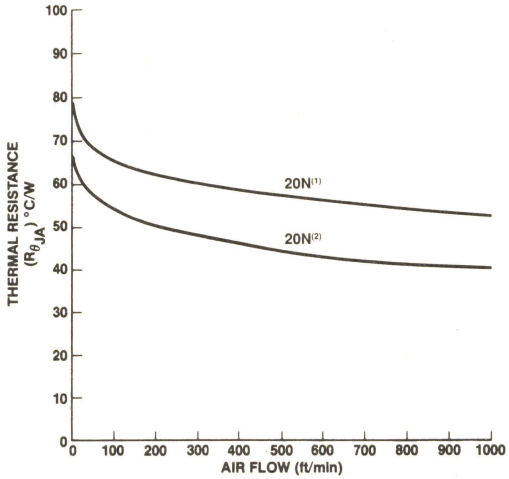
The thermal resistance measurement can be summarized as follows:

1. Calibration of the voltage drop across the sensing element with respect to temperature. This is done by measuring the voltage drop at several different temperatures with the heating power off.
2. Measurement of voltage drop across the sensing element under operating conditions, under various air flow rates (from 0 to 1000 linear ft/min.), while measuring °C ambient and power input for calculation of $R_{\theta JA}$.
3. Measurement of voltage drop across the sensing element under operating conditions, package immersed in constant temperature fluorinert bath, while measuring the case temperature at the bottom of the package and power input for calculation of $R_{\theta JC}$. The readings are recorded when the package has reached thermal equilibrium.
4. Calculation of thermal resistance

   a. $R_{\theta JA} = \frac{T_J - T_A}{P}$ b. $R_{\theta JC} = \frac{T_J - T_C}{P}$

3

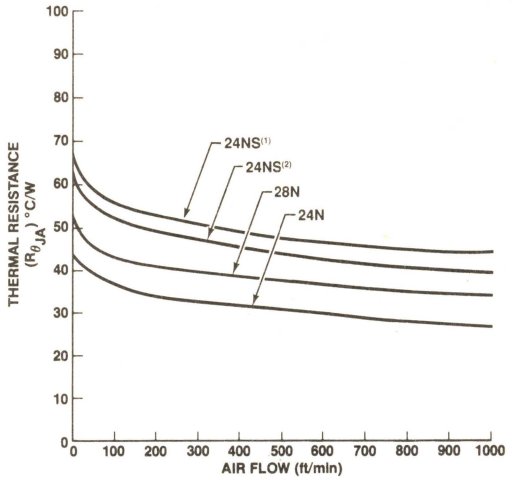
## 20 Lead Molded DIP (20N) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20N(1) | 5,625 | 22 |
| 20N(2) | 11,250 | 15 |

*These are typical values for the given die size.

## 24, 28 Lead Molded DIP (24N, 28N) Packages

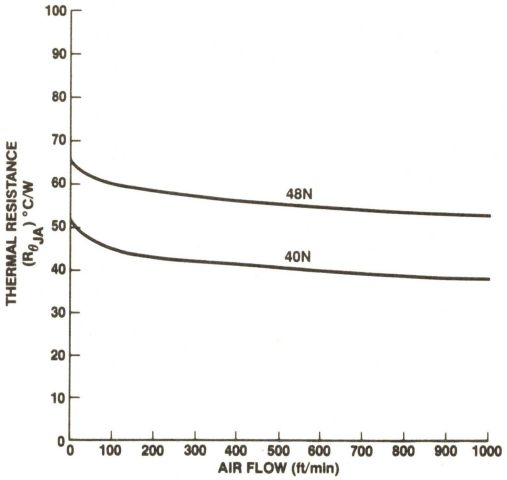



| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 24NS(1) | 5,625 | 20 |
| 24NS(2) | 11,250 | 15 |
| 24N | 50,625 | 10 |
| 28N | 22,500 | 13 |

*These are typical values for the given die size.

3

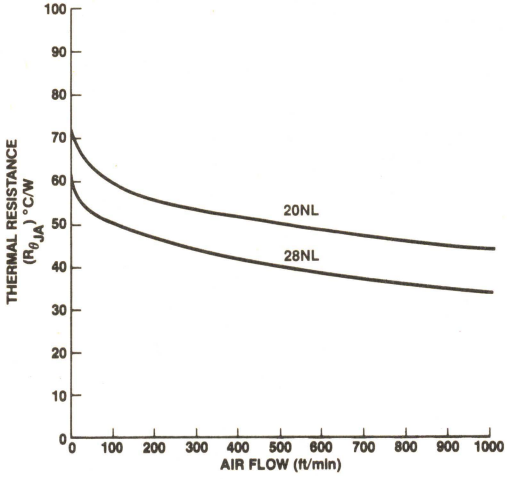
## 40, 48 Lead Molded DIP (40N, 48N) Packages

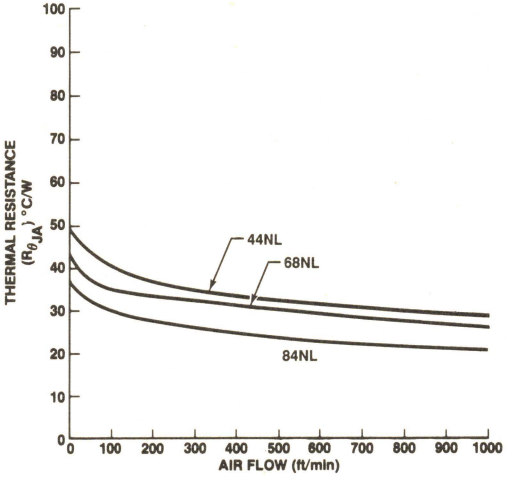



| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 40N | 22,500 | 16 |
| 48N | 5,625 | 23 |

*These are typical values for the given die size.

## Plastic Leaded Chip Carrier (NL) Packages

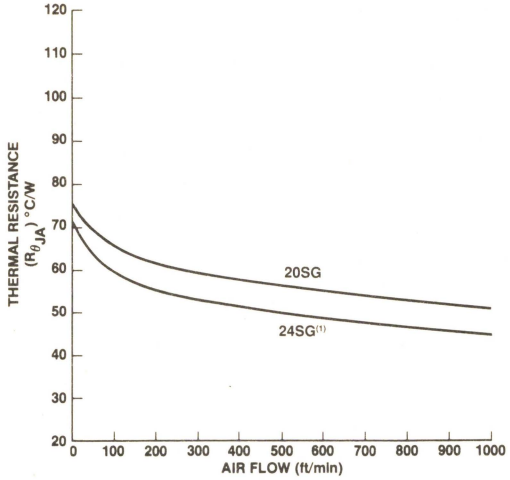



| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20NL | 11,250 | 14 |
| 28NL | 22,500 | 13 |
| 44NL | 22,500 | 11 |
| 68NL | 50,625 | 8 |
| 84NL | 50,625 | 6 |

*These are typical values for the given die size.

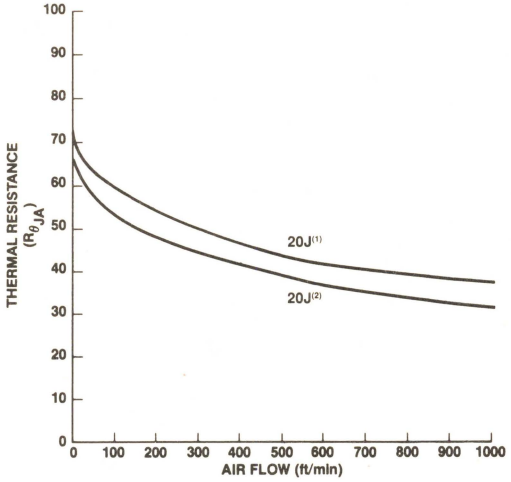
## Small Outline (20SG, 24SG) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20SG | 5,625 | 16 |
| 24SG(1) | 11,250 | 13 |
| 24SG(2) | 22,500 | 10 |

*These are typical values for the given die size.

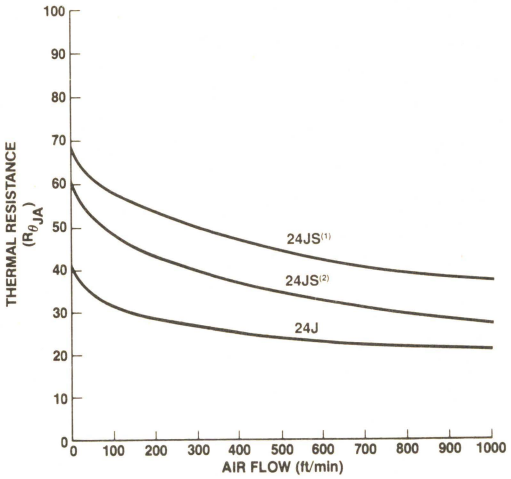
## 20 Lead Cerdip (20J) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20J(1) | 5,625 | 14 |
| 20J(2) | 22,500 | 6 |

*These are typical values for the given die size.

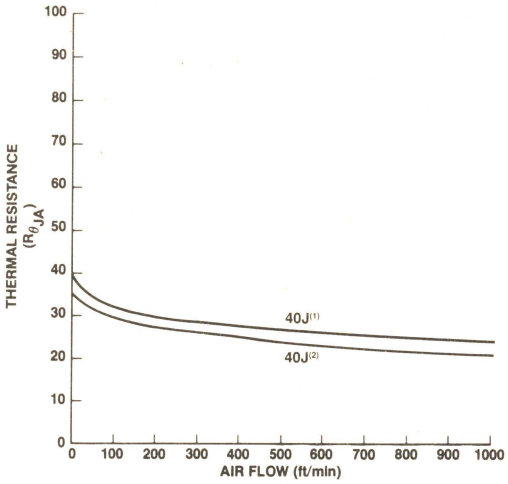
## 24 Lead Cerdip (24J, 24JS) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 24JS(1) | 5,625 | 16 |
| 24JS(2) | 11,250 | 8 |
| 24J | 50,625 | 3 |

*These are typical values for the given die size.

## 40 Lead Cerdip (40J) Packages

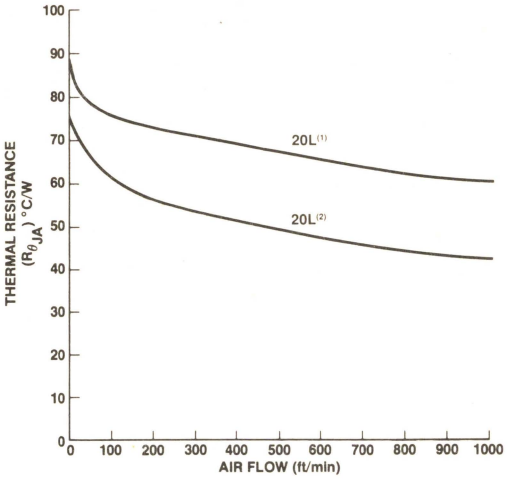



| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 40J(1) | 22,500 | 4 |
| 40J(2) | 50,625 | 2 |

*These are typical values for the given die size.

3

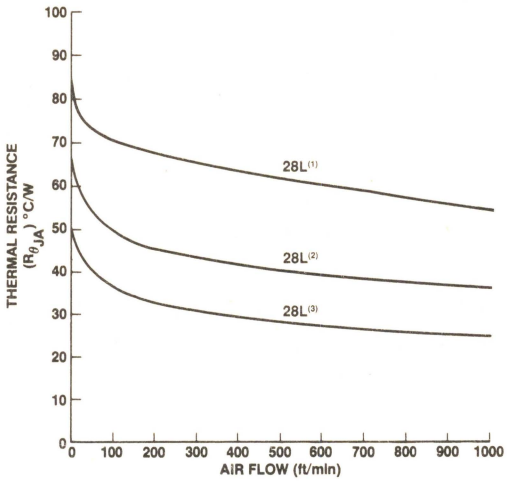
## 20 Leadless Chip Carrier (20L) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 20L(1) | 5,625 | 16 |
| 20L(2) | 22,500 | 4 |

*These are typical values for the given die size.

## 28 Leadless Chip Carrier (28L) Packages

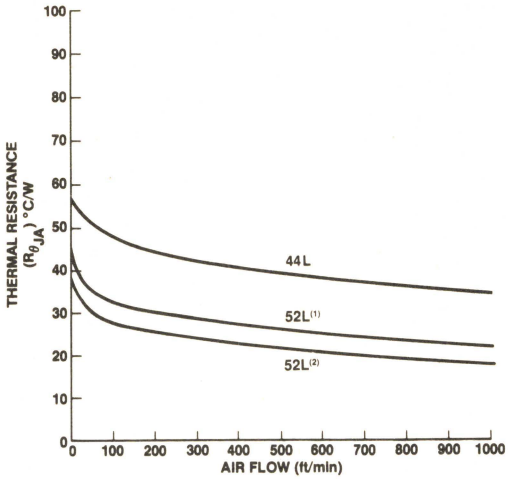



| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 28L(1) | 5,625 | 20 |
| 28L(2) | 11,250 | 10 |
| 28L(3) | 50,625 | 3 |

*These are typical values for the given die size.

3

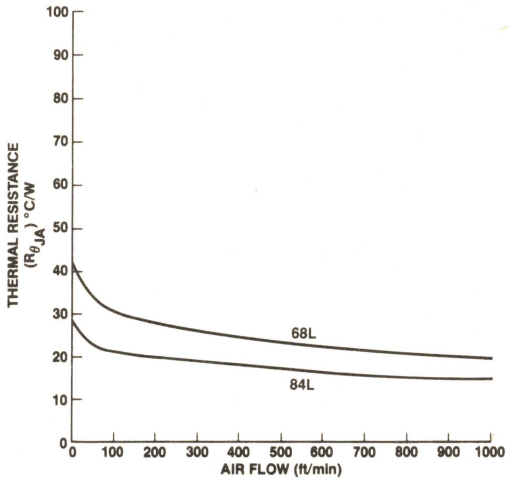
## Leadless Chip Carrier (44L, 52L) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 44L | 22,500 | 4 |
| 52L(1) | 22,500 | 2 |
| 52L(2) | 50,625 | 1.5 |

*These are typical values for the given die size.

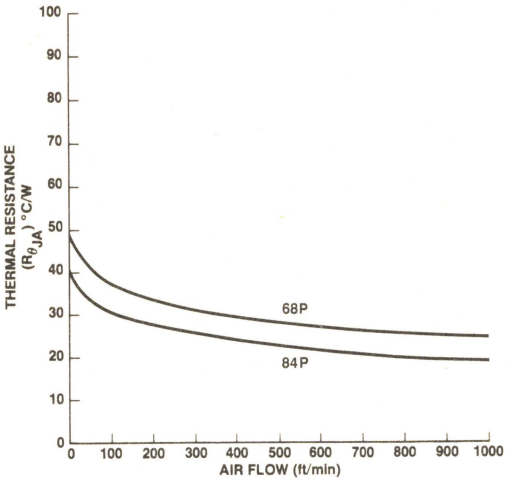
## Leadless Chip Carrier (68L, 84L) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 68L | 22,500 | 2 |
| 84L | 50,625 | 1 |

*These are typical values for the given die size.

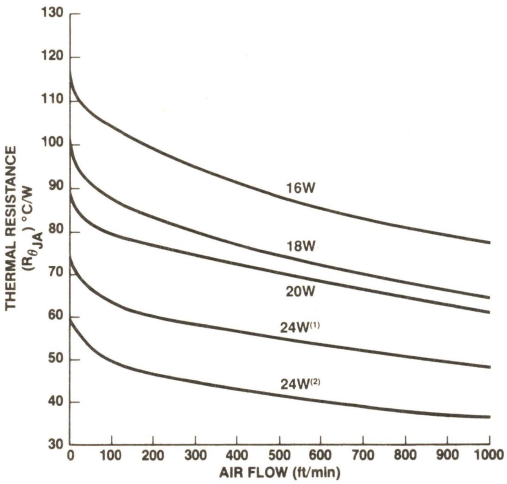
## Pin Grid Array (68P, 84P) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 68P | 22,500 | 5 |
| 84P | 90,000 | 2 |

*These are typical values for the given die size.

## Cerpack (W) Packages




| PACKAGE | DIE SIZE (mils)$^2$ | $R^*_{\theta JC}$ (°C/WATT) |
|---|---|---|
| 16W | 5,625 | 21 |
| 18W | 5,625 | 17 |
| 20W | 5,625 | 15 |
| 24W(1) | 11,500 | 7 |
| 24W(2) | 22,500 | 3 |

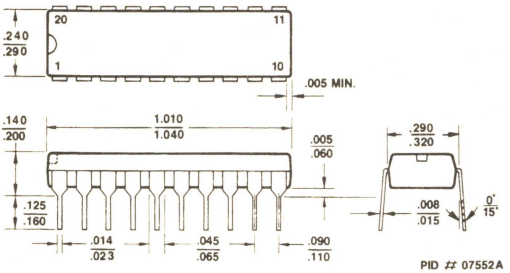
*These are typical values for the given die size.

3

# AmPAL Device Package Outlines

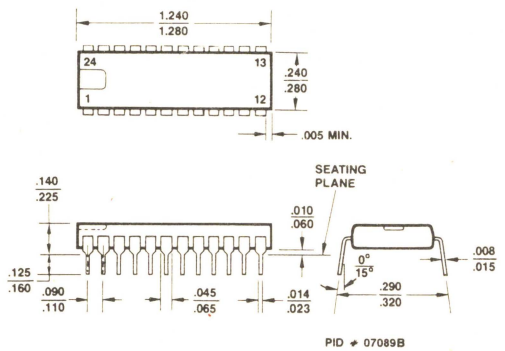
## Plastic Dual-In-Line Packages (PD)

### PD 020




PID # 07552A

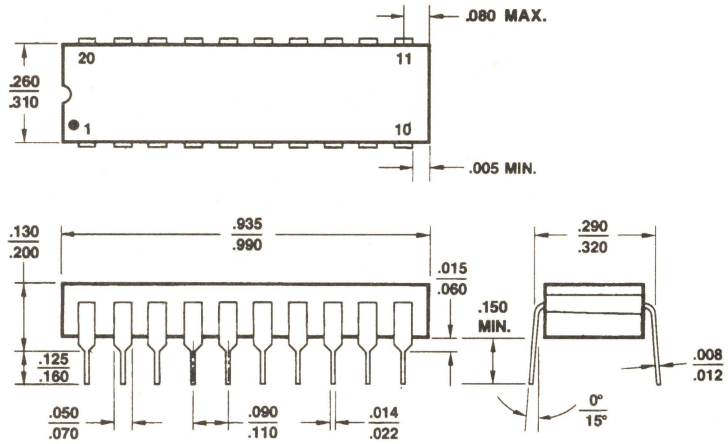
### PD3024




PID # 07089B

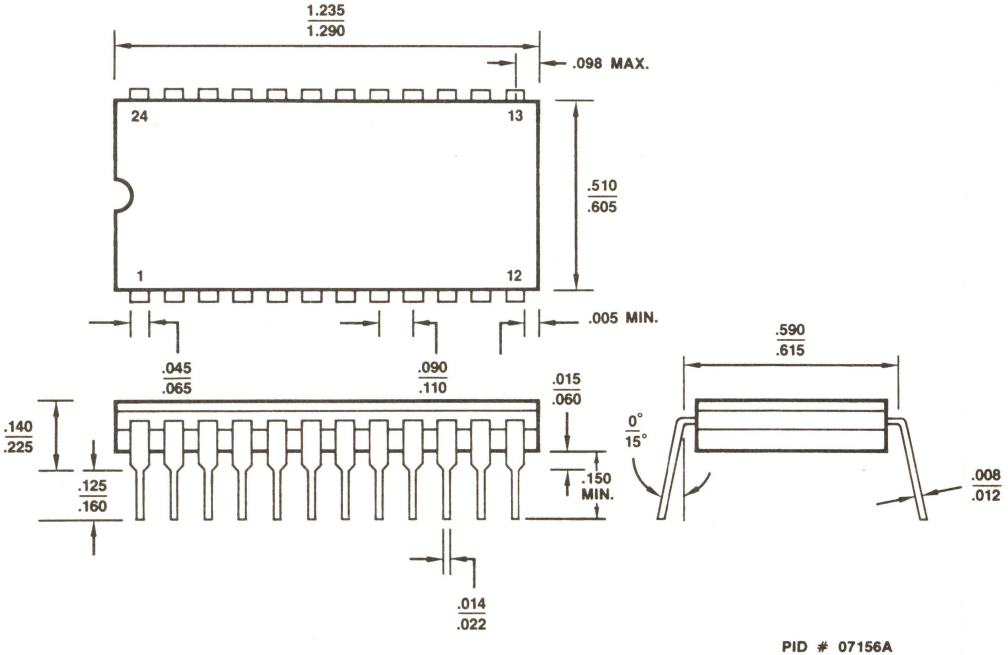
## Ceramic Hermetic Dual-In-Line Packages (CD)

### CD 020




PID # 07553A

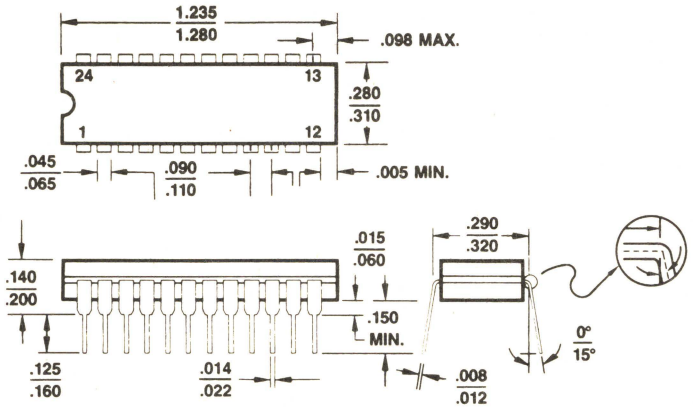
### CD 024




PID # 07156A

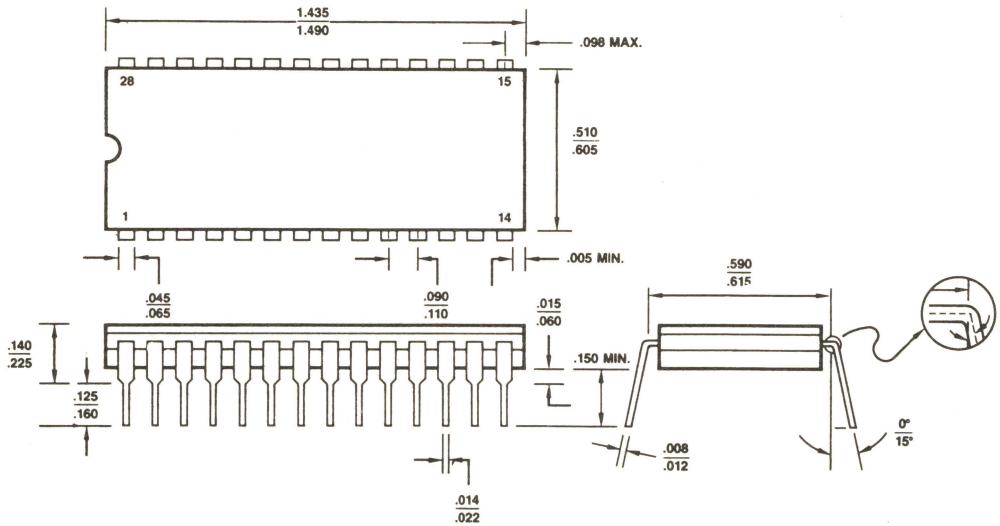
## Ceramic Hermetic Dual-In-Line Packages (CD) (Cont'd.)

### CD3024




PID # 06850B

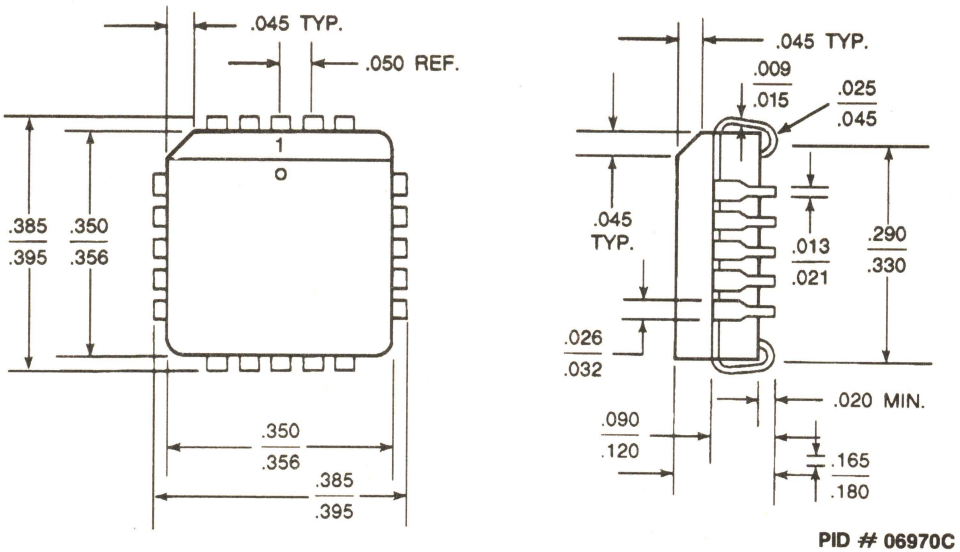
### CD 028




PID # 06837A

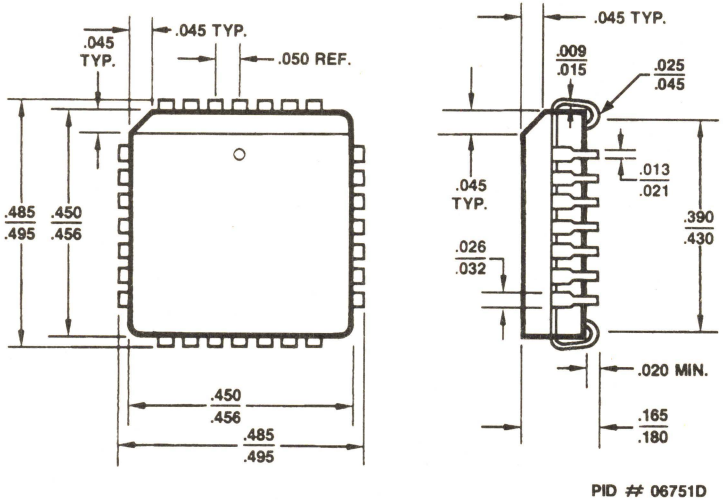
## Plastic Leaded Chip Carriers (PL)

### PL 020




PID # 06970C

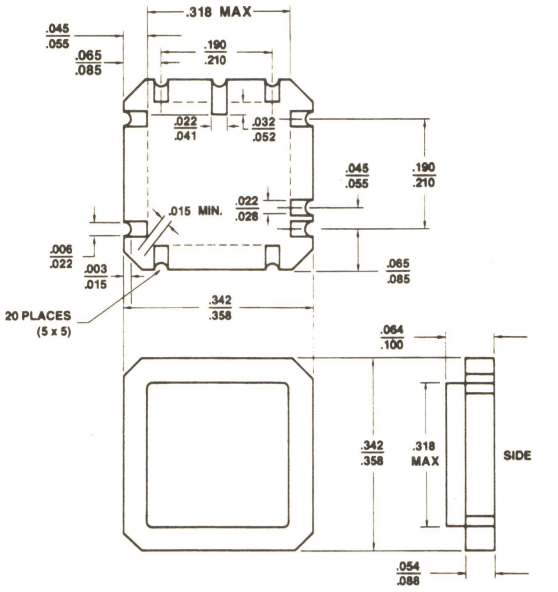
### PL 028




PID # 06751D

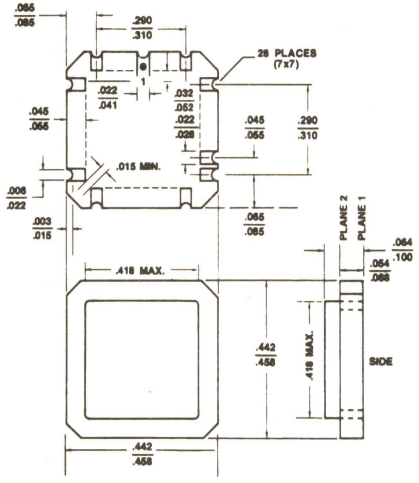
## Ceramic Leadless Chip Carriers (CL & CLR)

### CL 020




PID # 07318B

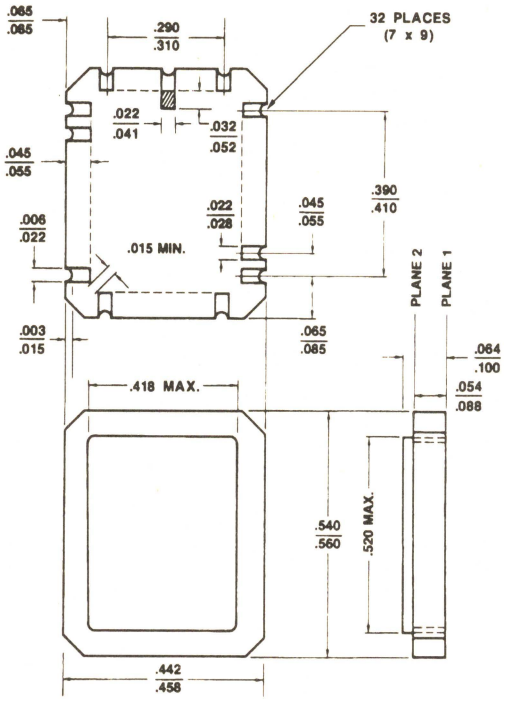
### CL 028




PID # 06595D

## Ceramic Leadless Chip Carriers (CL & CLR) (Cont'd.)

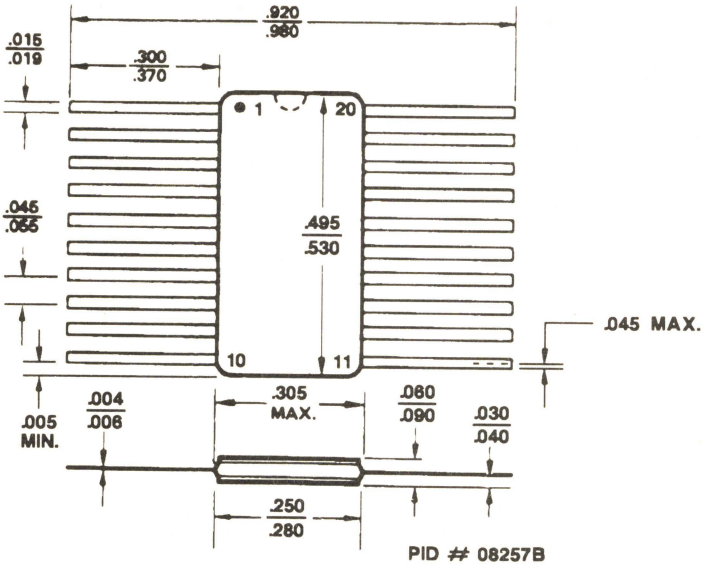
### CLR032



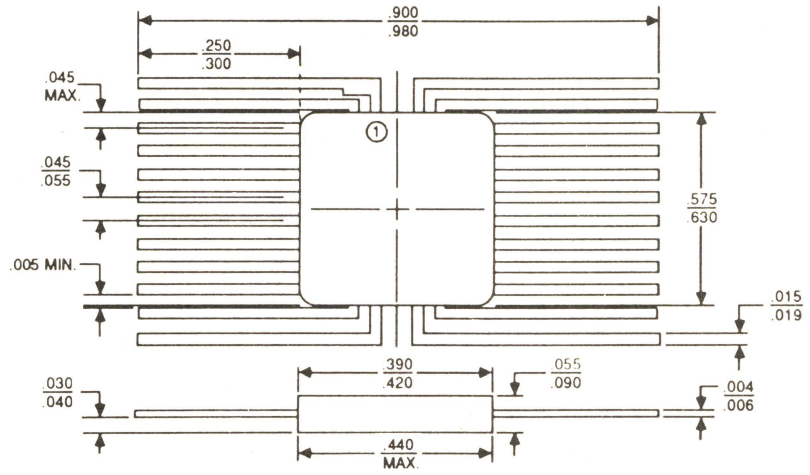

PID # 06841C

## Ceramic Flatpacks (CF)

### CF 020




### CFL 024



# AmPAL Device Package Thermal Characteristics

TR-202


## Abstract

Determination of the Thermal Resistance of Packaged Devices is of concern to the designer of new devices and to AMD customers. The Advanced Package and Material Development group has undertaken the task of characterizing current AMD products and quantifying package-related influences on Thermal Resistance. This report describes some of these effects and the technique used to measure Thermal Resistance.


## Definition of Thermal Resistance

The reliability of an integrated circuit is largely dependent on the maximum temperature which the device will attain during operation. Because the stability of a semiconductor junction declines with increasing temperature, knowledge of the thermal properties of the packaged device becomes an important factor during device design. In order to increase the operating lifetime of a given device, the junction temperatures must be minimized. This demands knowledge of the thermal resistance of the completed assembly and specification of the conditions in which the device will function properly. As devices become both smaller and more complex and the requirement for high speed operation becomes more important, heat dissipation will become an ever more critical parameter.

Thermal resistance is defined as the temperature rise per unit power dissipation above some referenced condition. The unit of measure is typically °C/watt. The relationship between junction temperature and thermal resistance is given by:

$$T_J = T_X + P_D\ \theta_{JX} \qquad (1)$$

where: $T_J$ = junction temperature
$T_X$ = reference temperature
$P_D$ = power dissipation
$\theta_{JX}$ = thermal resistance
X = some defined test condition

In general, one of three conditions is defined for measurement of thermal resistance:

$\theta_{JC}$ – thermal resistance measured with reference to the temperature at some specified point on the package surface.

$\theta_{JA}$ (still air) – thermal resistance measured with respect to the temperature of a specified volume of still air.

$\theta_{JA}$ (moving air) – thermal resistance measured with respect to the temperature of air moving at a specified velocity.

The relationship between $\theta_{JC}$ and $\theta_{JA}$ is

$$\theta_{JA} = \theta_{JC} + \theta_{CA}$$

where $\theta_{CA}$ is a measure of the heat dissipation due to natural convection (still air) or forced convection (moving air) and the effect of heat radiation and mounting techniques. $\theta_{JC}$ is dependent solely on material properties and package geometry; $\theta_{JA}$ includes the influence of the surface area of the package and environmental conditions. Each of these definitions of thermal resistance is an attempt to simulate some manner in which the package device may be used.

The thermal resistance of a packaged device, however measured, is a summation of the thermal resistances of the individual components of the assembly. These in turn are functions of the thermal conductivity of the component materials and the geometry of the heat flow paths. Like other material properties, thermal conductivity is usually temperature dependent. For alumina and silicon, two common package materials, this dependence can amount to a 30% variation in thermal conductivity over the operating temperature range of the device. The thermal resistance of a component is given by

$$\theta = \frac{L}{K(T)A} \qquad (2)$$

where: L = length of the heat flow path
A = cross sectional area of the heat flow path
K(T) = thermal conductivity as a function of temperature

and the overall thermal resistance of the assembly (discounting convective effects) will be:

$$\theta = \Sigma\theta_n = \Sigma\ \frac{L_n}{K_n A_n}$$

but since the heat flow path through a component is influenced by the materials surrounding it, determination of L and A is not always straightforward.

A second factor that affects the thermal resistance of a packaged device is the power dissipation level and, more particularly, the relationship between power level and die geometry, i.e., power distribution and power density. By rearrangement of equation 1 to

$$P_d = \frac{1}{\theta_{JX}}(T_J - T_X) = \frac{1}{\Sigma\theta_N}(T_J - T_X) \qquad (3)$$

the relationship between $P_d$ and $T_j$ can be more clearly seen. Thus, to dissipate a greater quantity of heat for a given geometry, $T_J$ must increase and, since the individual $\theta_n$ will also increase with temperature, the increase in $T_J$ will not be a linear function of increasing power levels.

A third factor of concern is the quality of the material interfaces. In terms of package construction, this relates



08765B/0
JANUARY 1988

specifically to the die attach bond, and for those packages having a heatsink, the heatsink attach bond. The quality of the die attach bond will most severely influence the package thermal resistance as this is the area which first impedes the transfer of heat out of the silicon die. Indeed, it seems likely that the initial thermal response of a powered device can be directly related to the quality of the die attach bond.

## Experimental Method

The technique for measurement of thermal resistance involves the identification of a temperature-sensitive parameter on the device and monitoring this parameter while the device is powered. For bipolar integrated circuits the forward voltage of the substrate isolation diode provides a convenient parameter to measure and has the advantage of a linear dependence on temperature. MOS devices which do not have an accessible substrate diode present greater measurement difficulties and may require simulation through use of a specially designed thermal test die. Choice of the parameter to be measured must be made with some care to ensure that the results of the measurement are truly representative of the thermal state of the device being investigated. Thus measurement of the substrate isolation diode which is generally diffused across the area of the die yields a weighted average of the condition of the individual junctions across the die surface. Measurement of a more local source would yield a less generalized result.

For MOS devices, simulation is accomlished using the thermal test die. The basis for this test die is a 25 mil square cell containing an isolated diode and a 1 KΩ resistor. The resistors are interconnected from cell to cell on the wafer before it is cut into mulitple arrays of the basic unit cell. In use the device is powered via the resistors with voltage or current adjusted for the proper level and the voltage drop of the individual diodes is monitored as in the case of actual devices.

Prior to the thermal resistance test, the diode voltage/temperature calibration must be determined. This is done by measuring the forward voltage at 1 mA current level at two different temperatures. The diode calibration factor is then:

$$K_f = \frac{T_2 - T_1}{V_2 - V_1} = \frac{\Delta T}{\Delta V} \qquad (4)$$

in units of °C/mV. For most diodes used for this test the voltage/temperature relationship is linear and these two measurement points are sufficient to determine the calibration.

The actual thermal resistance measurement has two alternating phases: measurement and power on. The device under test is pulse powered with an ON duty cycle of 99% and a repetition rate of $< 100$ Hz. During the brief OFF states the device is reverse-biased with a 1 mA current and the voltage drop is measured. The series of voltage readings are averaged over short periods and compared to the voltage reading obtained before the device was first powered ON. The thermal resistance is then computed as:

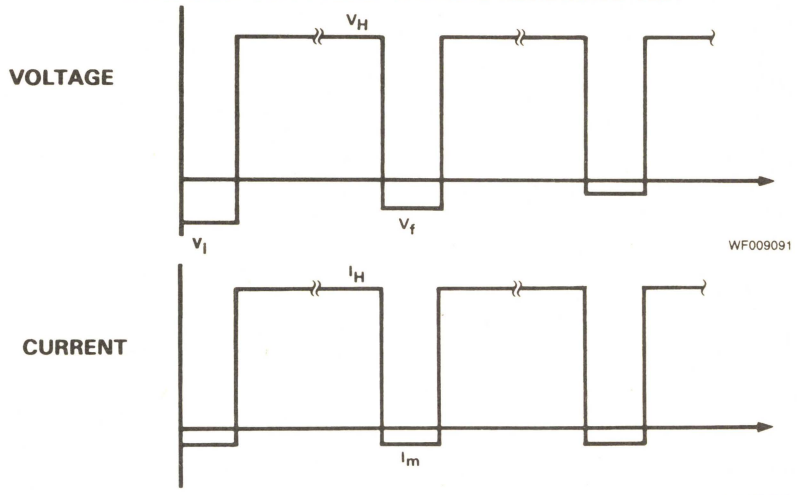
$$\theta_{jx} = \frac{K_F(V_F - V_I)}{V_H I_H} = \frac{K_f \Delta V}{P_D} \qquad (5)$$

where: $K_F$ = calibration factor
$V_I$ = initial forward voltage value
$V_F$ = current forward voltage value
$V_H$ = heating voltage
$I_H$ = heating current

The pulsing measurement is continued until the device has reached thermal equilibrium and the final value measured is the equilibrium thermal resistance of the device under test.

When the end result desired is $\theta_{JA}$ (still air), the device and the test fixture (typically a standard burn-in socket) are enclosed in a box containing approximately 1 cubic foot of air. For $\theta_{JC}$ measurements the device is attached to a large metal heatsink. This ensures that the reference point on the device surface is maintained at a constant temperature. The requirements for measurement of $\theta_{JA}$ (moving air) are rather more comple x and involve the use of a small wind tunnel with capability for monitoring air pressure, temperature and velocity in the area immediately surrounding the device tested. Standardization of this last test requires much careful attention.

**WAVEFORMS FOR PULSED THERMAL RESISTANCE TEST**



## Experimental Results

The thermal resistance data included in the attached table was extrapolated from data collected using the procedure outlined in the preceding section. This data has resulted from an ongoing program undertaken by members of the Material Technology Development group.

Updated data will replace the data in this table as each device is measured or revised data becomes available.

## Thermal Resistance of AMD Products

**(Notes 1, 2 and 3)**

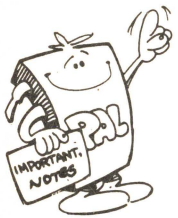
| PIN COUNT | PACKAGE TYPE (Note 4) | $\theta_{JA}$ | $\theta_{JC}$ |
|---|---|---|---|
| 20 | Ceramic DIP | 60 | 11 |
| | Plastic DIP | 61 | 30 |
| | Ceramic Flatpack | 56 | CR |
| | Ceramic LCC | 61 | CR |
| | Plastic LCC* | CR | CR |
| 24 | Ceramic DIP | 57 | 15 |
| | Plastic DIP | 60 | CR |
| | Ceramic Flatpack | 85 | 9 |
| 28 | Ceramic LCC | CR | CR |
| | Plastic LCC* | 58 | CR |

Notes:

1. Representative values for each package type — for information only.
2. Any given device may differ from these values. Consult local AMD sales office for specific-device information.
3. CR = Consult local AMD Representative.
4. DIP = Dual-In-Line Package
   LCC = Leadless Chip Carrier
   LCC* = Leaded Chip Carrier

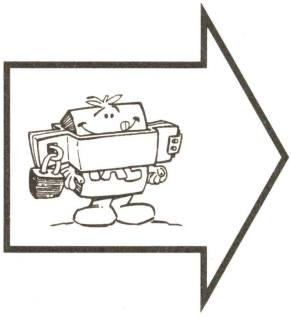
**Table 1.**

3


IMPORTANT
NOTES

## PAL Device Handbook


| | |
|---|---|
| Introduction | 1 |
| Applications | 2 |

## PAL Device Data Book


| | |
|---|---|
| Programming and Quality | 3 |
| PALASM 2 Software User Documentation | 4 |
| Data Sheets | 5 |
| Appendices | 6 |

Introduction ..... 4-1

Install PALASM 2 Software ..... 4-15

Run the Software ..... 4-29

Build a Boolean Equation Design ..... 4-61

Build a State Machine Design ..... 4-95

Build Simulation ..... 4-137

Program the Device ..... 4-169

PALASM 2 Software Glossary ..... 4-183

PALASM 2 Software Index ..... 4-189

(Detailed Table of Contents on page 4-v)

# Preface

## *Audience*

The intended audience for this manual is design engineers who run PALASM® 2 software on an IBM-PC/XT/AT computer to program Monolithic Memories programmable logic devices (PLDs). Users of this manual should be familiar with PLD technology and with PLD programming concepts. If you are a new user, refer to the introductory chapters of this handbook for background information and tutorials, including the Beginner's Guide.

## *Using This Manual*

This chapter of the handbook is a self-contained manual that describes step-by-step instructions for installing PALASM 2 software and programming a Monolithic Memories programmable logic device. As a self-contained PALASM manual, the information is organized into chapters (numbered 1 to 7) that include sections and subsections. The handbook chapter designator, 4-, precedes all page, figure, and table numbers, which are numbered consecutively throughout the handbook.

Throughout this manual, note these conventions:

- *Italic* typeface indicates references to other sections or chapters.
- `Courier` typeface represents information displayed on a computer screen.
- Left and right arrows < > indicate keys on the computer keyboard. For example, <F1>, <Y>, or <esc>.
- <return> represents the carriage return key. Some keyboards label this key as Ret, enter, or an arrow pointing left.

Figure 4-1 shows the structure of PALASM 2 Software User Documentation.

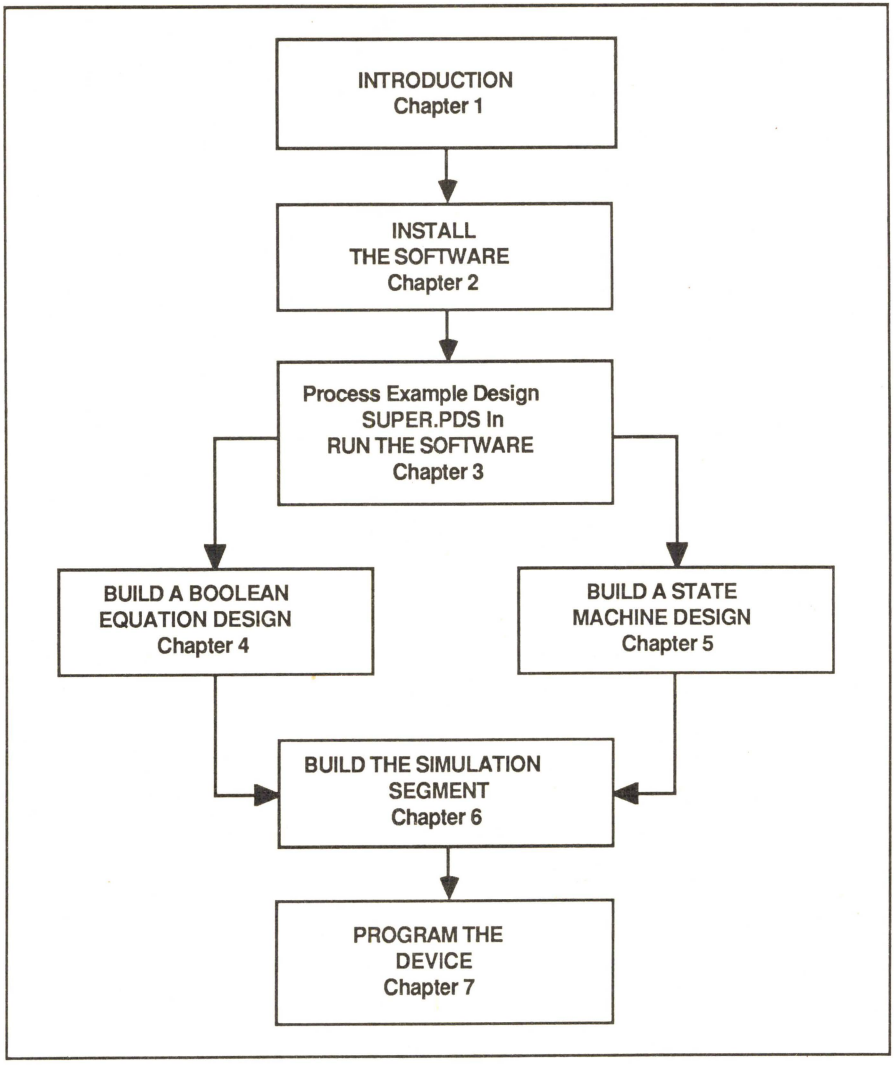



Figure 4-1

Structure of PALASM 2 Software User Documentation

## Other Documents

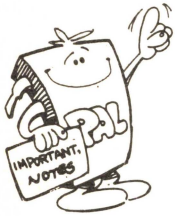
Make sure you have available the user manuals for your specific computer and PLD programmer. You will refer to these manuals when installing the PALASM 2 software and programming a device.

For more information about programmable logic and PAL devices, refer to the first three chapters of this handbook.

Documentation on PLPL, AMD's programmable logic software, can also be found in this handbook. PLPL software is included in the PALASM 2 software package.

## Where To Get Help

Monolithic Memories maintains an applications hotline to help you solve engineering-related problems. If you have trouble installing or running PALASM 2 software, call the hotline at 800-222-9323.


IMPORTANT
NOTES

# Table of Contents


Preface ..... 4-i
List Of Figures ..... 4-ix
List Of Tables ..... 4-xi

## CHAPTER 1
## INTRODUCTION

1.1 Introducing PALASM 2 Software ..... 4-2
1.1.1 Supported Programmable Logic Devices ..... 4-2
1.1.2 Supported Computers ..... 4-4
1.2 Program and File Summary ..... 4-5
1.2.1 PALASM 2 Software Programs ..... 4-6
1.2.2 Input, Output And Intermediate Files ..... 4-11
1.2.3 PALASM 2 Supplementary Programs ..... 4-12

## CHAPTER 2
## INSTALL PALASM 2 SOFTWARE

2.1 Installation Procedures ..... 4-16
2.1.1 What You Require ..... 4-16
2.1.2 Install PALASM 2 Software With Two Floppy Disk Drives ..... 4-16
2.1.3 Install PALASM 2 Software On A Hard Disk Drive ..... 4-18
2.2 Software Setup ..... 4-23
2.3 Add Supplementary Programs To The Menu ..... 4-25
2.4 Modifications To The AUTOEXEC.BAT And CONFIG.SYS Files ..... 4-28

## CHAPTER 3
## RUN THE SOFTWARE

3.1 Overview Of The Procedure ..... 4-30
3.2 Prepare To Use The Menu ..... 4-32
3.2.1 Call The PALASM Menu ..... 4-32
3.2.2 Specify The Directory And Input File ..... 4-35
3.3 Open The Sample Input File ..... 4-36

3.4 Study The Sample Input File ..... 4-36
3.5 Autorun Assembly And Simulation ..... 4-36
3.6 Process The Input File ..... 4-38
3.6.1 Check The Syntax Of The Input File ..... 4-38
3.6.2 Expand The Input Equations ..... 4-40
3.6.3 Minimize The Input Equations ..... 4-41
3.6.4 Assemble The Input File ..... 4-42
3.6.5 View The Assembly Output Files ..... 4-42
3.7 Simulate The Sample Design ..... 4-46
3.7.1 Run The Simulation Program ..... 4-46
3.7.2 View The Simulation Output Files ..... 4-47
3.7.3 View The JEDEC Test Data ..... 4-50
3.8 Disassemble A JEDEC File ..... 4-50
3.9 Identify Errors In The Input File ..... 4-52
3.9.1 View The Run-Time Log ..... 4-53
3.9.2 Disassemble The TRE File ..... 4-54
3.10 Interpret The Assembly Output Files ..... 4-56
3.10.1 Interpret The Fuseplot ..... 4-56
3.10.2 View The JEDEC File ..... 4-57
3.11 Run The Software From DOS ..... 4-59


## CHAPTER 4
## BUILD A BOOLEAN EQUATION DESIGN


4.1 Build A Boolean Equation Design ..... 4-62
4.1.1 General Syntax ..... 4-63
4.1.2 Build The Declaration Segment ..... 4-65
4.1.3 Build The Equations Segment ..... 4-70
4.2 Polarity ..... 4-76
4.2.1 Programmable Polarity ..... 4-76
4.2.2 Fixed Polarity ..... 4-80
4.3 Tailor The Design For Specific Devices ..... 4-80
4.3.1 PLS Device General Considerations ..... 4-81
4.3.2 PAL Device General Considerations ..... 4-83
4.3.3 PAL10H20G8 And PAL10H20GR8 Special Considerations ..... 4-83
4.3.4 PAL22V10 Special Considerations ..... 4-84
4.3.5 PAL16RA8 And PAL20RA10 Special Considerations ..... 4-85
4.3.6 PAL32R16 And PAL64R32 Special Considerations ..... 4-87
4.3.7 PAL32VX10 Special Considerations ..... 4-87
4.4 Checklist For Verifying Boolean Equation Designs ..... 4-92

## CHAPTER 5
## BUILD A STATE MACHINE DESIGN


5.1 Create A State Diagram ..... 4-96
5.1.1 Create A Mealy State Diagram ..... 4-96
5.1.2 Create A Moore State Diagram ..... 4-100
5.2 Build A State Machine Design ..... 4-103
5.2.1 General Syntax ..... 4-105
5.2.2 Build The Declaration Segment ..... 4-107
5.2.3 Build The State Segment ..... 4-112
5.2.4 Build The Conditions Segment ..... 4-125
5.3 Tailor The Design For PLS, PROSE, Or PAL Device State Machines ..... 4-127
5.3.1 PLS And PROSE Considerations ..... 4-127
5.3.2 PAL Device General Considerations ..... 4-130
5.4 Review A Simple Design ..... 4-132


## CHAPTER 6
## BUILD SIMULATION


6.1 Special Syntax ..... 4-138
6.2 Build The Simulation Segment ..... 4-138
6.2.1 The Simulation Language ..... 4-139
6.2.2 Review Simulation Guidelines ..... 4-150
6.2.3 Rules For State Machine Simulation Syntax ..... 4-151
6.3 Review A Sample Design And Interpret The Output Files ..... 4-151
6.3.1 Interpret The History Waveforms ..... 4-157
6.3.2 Interpret The Trace Waveforms ..... 4-159
6.3.3 Interpret The History File ..... 4-161
6.3.4 Interpret A PROSE History File ..... 4-163
6.3.5 Interpret The Trace File ..... 4-165
6.3.6 Interpret The JEDEC Test Data ..... 4-166


## CHAPTER 7
## PROGRAM THE DEVICE


7.1 Send JEDEC Files To The Programmer ..... 4-170
7.1.1 Connect the Programmer ..... 4-170
7.1.2 Set Up The Communications Link ..... 4-172
7.1.3 Transmit The JEDEC File Using MS-DOS ..... 4-173
7.2 PC2 Communications Software ..... 4-173
7.2.1 Load PC2 ..... 4-174
7.2.2 Set Up Computer Transmission Parameters ..... 4-176

7.2.3 Transmit The JEDEC File ..... 4-178
7.3 Copy From A Master Device ..... 4-180
7.4 Program The Device ..... 4-182

PALASM 2 SOFTWARE GLOSSARY ..... 4-183

PALASM 2 SOFTWARE INDEX ..... 4-189

# LIST OF FIGURES


| | | |
|---|---|---|
| 4-1 | Structure of PALASM 2 Software User Documentation | 4-ii |
| 4-2 | Typical Computer Configuration | 4-5 |
| 4-3 | PALASM 2 Software Flow for PAL Devices | 4-6 |
| 4-4 | PALASM 2 Software Main Menu | 4-18 |
| 4-5 | PALASM Input Install Request Menu | 4-20 |
| 4-6 | PALASM 2 Software Installation Menu | 4-25 |
| 4-7 | MENU.SYS | 4-27 |
| 4-8 | PALASM 2 Software Processing Sequence | 4-30 |
| 4-9 | Basic Design Options | 4-31 |
| 4-10 | The Main Menu | 4-34 |
| 4-11 | Page One of the PALASM2 Sub-menu | 4-39 |
| 4-12 | The Syntax Check Operation | 4-40 |
| 4-13 | Assembly and Simulation Output Files | 4-43 |
| 4-14 | Page Two of the View Data Sub-menu | 4-45 |
| 4-15 | Page One of the View Data Sub-menu | 4-48 |
| 4-16 | History Waveforms | 4-49 |
| 4-17 | Page Two of the PALASM2 Sub-menu | 4-51 |
| 4-18 | Disassemble The TRE File | 4-54 |
| 4-19 | The SUPER.XPT Fuse Map | 4-57 |
| 4-20 | The JEDEC Fuse Data from SUPER.JED | 4-58 |
| 4-21 | Structure of the Boolean Equation Design | 4-62 |
| 4-22 | Structure of the Declaration Segment | 4-66 |
| 4-23 | CHIP Syntax and Pin List | 4-67 |
| 4-24 | STRING Information and Syntax | 4-68 |
| 4-25 | Sample Declaration Segment | 4-69 |
| 4-26 | Sample CHIP Entry and Combinatorial Equation | 4-71 |
| 4-27 | Sample CHIP Entry and Registered Equation | 4-73 |
| 4-28 | Comparison of Polarity in Pin List and Equations | 4-77 |
| 4-29 | Summary of Output Polarity for Programmable Polarity Parts | 4-79 |
| 4-29A | The PAL32VX10 Macrocell | 4-90 |
| 4-30 | Mealy Output | 4-96 |
| 4-31 | Mealy Functional State Diagram | 4-97 |
| 4-32 | Inputs and Outputs for Figure 4-31 | 4-98 |
| 4-33 | Minimum Inputs and Outputs to Build a Mealy Design | 4-99 |
| 4-34 | Moore Output | 4-100 |
| 4-35 | Moore Functional State Diagram | 4-101 |
| 4-36 | Inputs and Outputs in Moore Diagram | 4-102 |
| 4-37 | Minimum Inputs and Outputs to Build a Moore Design | 4-103 |
| 4-38 | Structure of the Design | 4-104 |
| 4-39 | Structure of the Declaration Segment | 4-108 |
| 4-40 | CHIP Syntax and Pin List | 4-109 |
| 4-41 | STRING Information and Syntax | 4-110 |
| 4-42 | Sample Declaration Segment | 4-112 |
| 4-43 | Structure of the State Segment | 4-113 |

LIST OF FIGURES (Continued)


4-44 Sample Defaults ..... 4-116
4-45 Sample State Assignments ..... 4-117
4-46 State Equation for Mealy or Moore Machine ..... 4-119
4-47 Transition and Output Equations for Mealy Machine ..... 4-122
4-48 Sample Transition and Output Equations ..... 4-123
4-49 Sample State Segment ..... 4-124
4-50 Location of Conditions Segment ..... 4-125
4-51 State Diagram of Conflicting Conditions ..... 4-126
4-52 Sample Conditions Segment ..... 4-127
4-53 Simple State Diagram ..... 4-133
4-54 Design for Figure 4-53 ..... 4-134
4-55 Location of the Simulation Segment ..... 4-139
4-56 SUPER.TRF Input File ..... 4-152
4-57 Waveform Display and Output File Format ..... 4-155
4-58 SUPER.HST Waveforms ..... 4-158
4-59 SUPER.TRF Waveforms ..... 4-160
4-60 Sample of SUPER.HST File ..... 4-162
4-61 Sample PMS14R21 History File ..... 4-164
4-62 Sample Trace File ..... 4-165
4-63 Test Vectors from SUPER.JDC ..... 4-167
4-64 Connect the Programmer to a Computer ..... 4-171
4-65 PC2 Function Key Menu Screen ..... 4-175
4-66 Computer Transmission Parameters Screen ..... 4-177
4-67 PC2 Name Transmit File Screen ..... 4-179
4-68 PC2 Exit Screen ..... 4-181

# LIST OF TABLES


| | | |
|---|---|---|
| 4-1 | PLDs Supported by PALASM 2 Software | 4-3 |
| 4-2 | PALASM 2 Software Programs | 4-7 |
| 4-3 | Input, Output, and Intermediate Files | 4-11 |
| 4-4 | PALASM 2 Software Supplementary Programs | 4-12 |
| 4-5 | Additional Supplementary Programs | 4-26 |
| 4-6 | Description of Boolean Equation Design Segments | 4-62 |
| 4-7 | Special Characters and Functions | 4-63 |
| 4-8 | Compilation of String Definitions in Figure 4-24 | 4-69 |
| 4-9 | Summary of Signals for Active-Low Output | 4-78 |
| 4-10 | Table for Determining Output Polarity | 4-78 |
| 4-11 | Syntax for PAL16RA8 and PAL20RA10 Functional Equations | 4-86 |
| 4-12 | Polarity Options for Output/Feedback and Register/Feedback Paths for Signal S2 | 4-90 |
| 4-13 | Options for Path Polarities and Specifying Output Pin Polarity for Signal O and Register Polarity for Signal R | 4-91 |
| 4-14 | Description of State Machine Design Segments | 4-104 |
| 4-15 | Special Characters and Functions | 4-105 |
| 4-16 | Compilation of String Definitions in Figure 4-41 | 4-111 |
| 4-17 | Descriptions of State Information | 4-113 |
| 4-18 | Descriptions of Default Options | 4-114 |
| 4-19 | Considerations for Tailoring State Machine Design Files for PLS and PROSE Devices | 4-128 |
| 4-20 | Special Features and Considerations for Specific PAL Devices | 4-131 |
| 4-21 | Exceptions to General Syntax | 4-138 |
| 4-22 | Description of Simulation Commands | 4-140 |
| 4-23 | Table for Using SETF to Control a Dedicated Output Enable | 4-143 |
| 4-24 | Table for Checking Signals | 4-146 |
| 4-25 | Simulation Output Characters | 4-156 |
| 4-26 | Value Characters in the History File | 4-163 |
| 4-27 | PC2 Function Keys | 4-176 |


IMPORTANT
NOTES

# 1. Introduction

### *About This Chapter*

Read this chapter before you use PALASM 2 software to get an overview of the features, functions, and software processing sequence.

| *For a description of...* | *Refer to Section...* |
|---|---|
| Supported programmable logic devices and computer environments | 1.1 |
| Programs, supplementary programs, input and output files that make up PALASM 2 software | 1.2 |

1.1

## Introducing PALASM 2 Software

PALASM 2 software uses the PLD design you create as an input file and converts it into a JEDEC file that can be used to program programmable logic devices (PLDs) on a programmer. The design you create specifies the fuses to be programmed on a device. PALASM 2 software accepts designs in Boolean or state equations. Your design can also include simulation guidelines that allow you to test your design without actually programming a device. PALASM 2 software accepts the design as input and performs a number of functions under your control. You can:

- Check the syntax of the input file
- Assemble the file
- Generate PLD fuse patterns in JEDEC format
- Report errors in syntax and assembly
- Simulate the PLD design

1.1.1

### Supported Programmable Logic Devices

With the exception of the PAL16A4 and the PAL16X4 parts, PALASM 2 software supports all Monolithic Memories programmable logic devices, including new PAL products such as RA (Registered Asynchronous), RS (Registered Synchronous), MegaPAL, ZHAL™, the PROSE device, PMS14R21, and the new PLS family of devices.

Table 4-1 lists the PLDs supported by PALASM 2 software.

Table 4-1

PLDs Supported by PALASM 2 Software

| 20-Pin PAL Devices | 24-Pin PAL Devices | MegaPAL Devices | PROSE Devices | PLS Devices |
|---|---|---|---|---|
| PAL10H8 | PAL6L16 | PAL32R16 | PMS14R21 | PLS105 |
| PAL10L8 | PAL8L14 | PAL64R32 | | PLS167 |
| PAL12H6 | PAL12L10 | | | PLS168 |
| PAL12L6 | PAL14L8 | | | |
| PAL14H4 | PAL16L6 | | | |
| PAL14L4 | PAL18L4 | | | |
| PAL16H2 | PAL20L2 | | | |
| PAL16L2 | PAL20C1 | | | |
| PAL16L8 | PAL20L8 | | | |
| PAL16P8 | PAL20L10 | | | |
| PAL16C1 | PAL20X4 | | | |
| PAL16R4 | PAL20X8 | | | |
| PAL16R6 | PAL20X10 | | | |
| PAL16R8 | PAL20R4 | | | |
| PAL16RA8 | PAL20R6 | | | |
| PAL16RP4 | PAL20R8 | | | |
| PAL16RP6 | PAL20RA10 | | | |
| PAL16RP8 | PAL20S10 | | | |
| PAL18P8 | PAL20RS4 | | | |
| ZHAL20 | PAL20RS8 | | | |
| | PAL20RS10 | | | |
| | PAL22V10 | | | |
| | PAL10H20P8 | | | |
| | PAL10H20G8 | | | |
| | PAL32VX10 | | | |
| | PAL22RX8 | | | |
| | ZHAL24 | | | |

1.1.2

## Supported Computers

PALASM 2 software operates with no user modification in the following computer environments. Monolithic Memories provides PALASM 2 software as an executable program, ready to run on any of these systems:

| | |
|---|---|
| Minicomputers: | VAX™ under VMS™<br>VAX™ under UNIX™ (Berkeley 4.2) |
| Microcomputers: | IBM-PC™, -XT™, -AT™<br>under MS-DOS™ (384K RAM) |
| Workstations: | DAISY™ under DNIX™ 5.1 |

This manual documents the installation and operation procedures for an IBM microcomputer environment. If your system is not an IBM-PC/X/AT, your PALASM 2 software package includes the installation and operation procedures for your particular system. The other information in this manual (Chapters 1, 4-7) applies to all environments.

**Note**: You should equip floppy-based systems with two disk drives.

**Note:** Refer to the PALASM 2 software ordering procedure included in this handbook for the correct part number of the software for your computer environment.

Your system must have a serial port (RS-232) if you need to communicate with the PLD programmer. *Program the Device*, Chapter 7, describes how to connect your computer to the programmer to download the JEDEC file. Figure 4-2 shows a typical computer configuration.

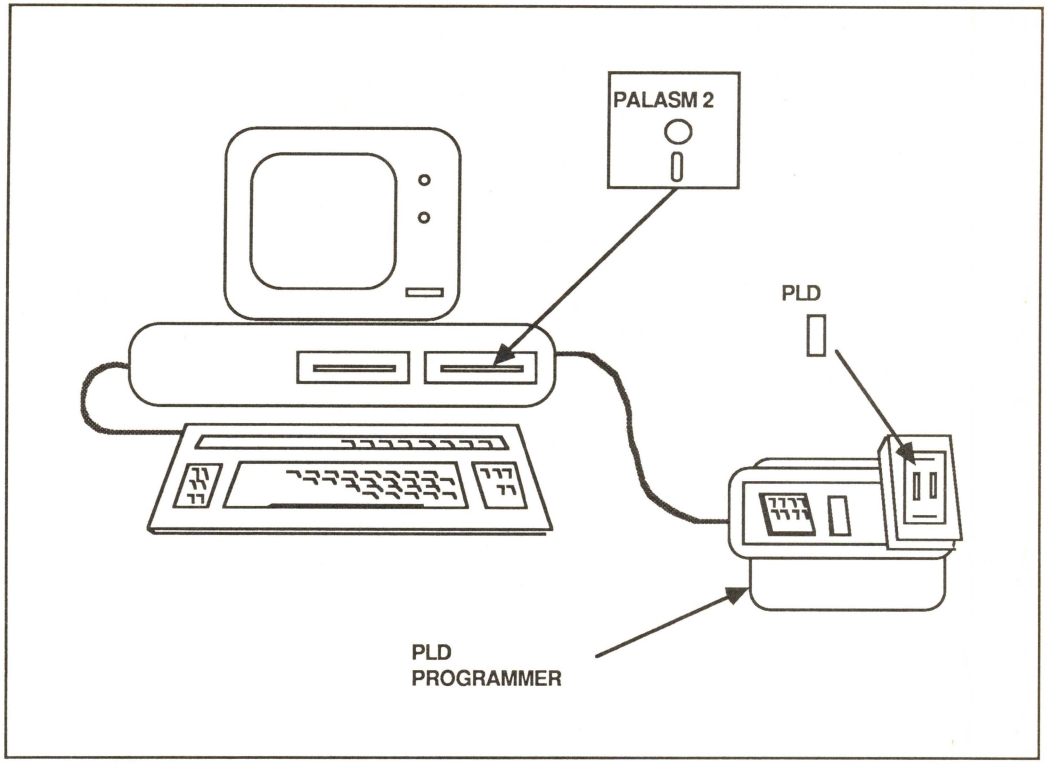



Figure 4-2

Typical Computer Configuration



1.2

## Program and File Summary

This section lists the current PALASM 2 software programs, followed by a brief description of each program. Input, intermediate, and output files are listed after the program summaries. Finally, the supplementary programs distributed with the software are briefly described. Figure 4-3 illustrates the sequence in which the software processing occurs.

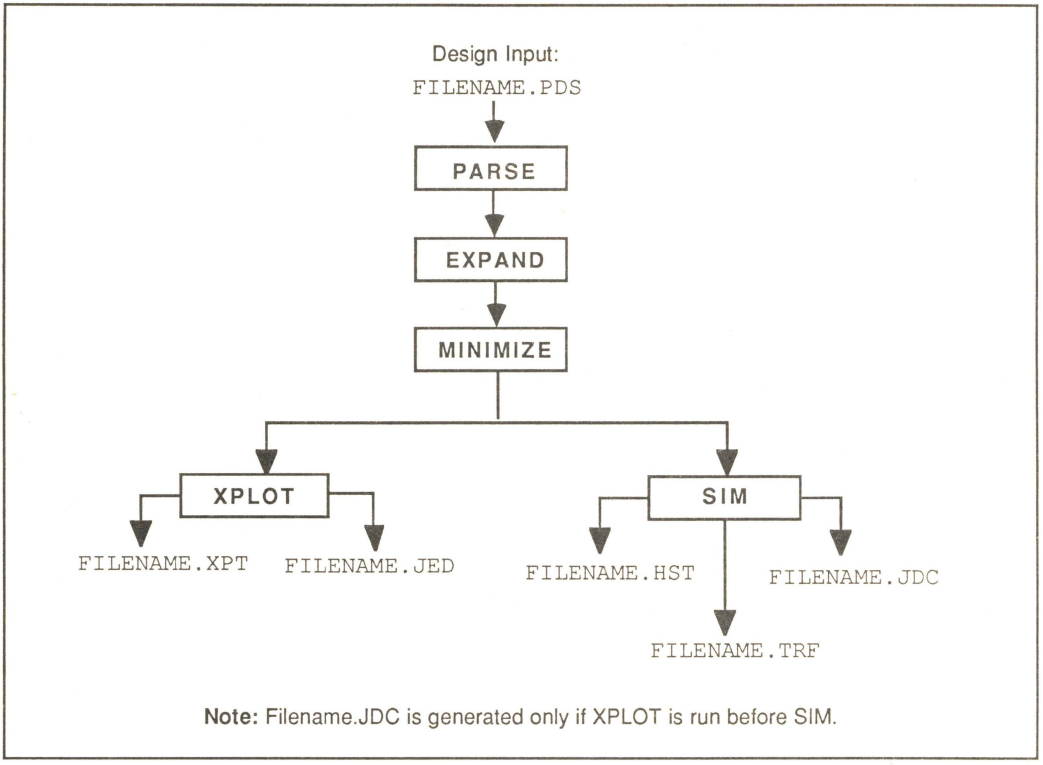



**Figure 4-3**

**PALASM 2 Software Flow for PAL Devices**

**Note:** On a PROSE device, the software substitutes XPLOT with PROASM and SIM with PROSIM. On a PLS device, the software substitutes XPLOT with PLSASM. PALASM 2 software currently includes the programs listed in Table 4-2.

1.2.1

## PALASM 2 Software Programs

Each program in the PALASM 2 software package is described in the following sections. Figure 4-3 shows the PALASM 2 software processing sequence for PAL devices along with the input and output files. Section 1.2.2 describes these files.

Table 4-2

PALASM 2 Software Programs

| Program | Function |
|---|---|
| PALASM | PALASM 2 software menu program |
| PARSE | Checks the syntax of the input file |
| EXPAND | Expands input equations and converts state machine syntax to Boolean equations |
| MINIMIZE | Minimizes equations |
| XPLOT | Assembles PAL device designs |
| SIM | Simulates PAL device designs |
| PROASM-PROSIM | Assembles and simulates PROSE device designs |
| PLSASM | Assembles PLS device designs |
| JEDMAN | Disassembles JEDEC files to Boolean equation input files |
| TREPL2 | Disassembles intermediate files created by PARSE, EXPAND and MINIMIZE. |



For a detailed description of each program, proceed to *PALASM 2 Software Programs*, Section 1.2.1. For a description of the supplementary programs, proceed to *PALASM 2 Supplementary Programs*, Section 1.2.2.

1.2.1.1

### PALASM

PALASM is the interactive menu program that simplifies user interface to the software. You can install PALASM on an IBM-PC/XT/AT, with either a twin floppy system or a hard disk drive. The user-friendly menu screens display all your options on one screen, enable the use of function keys to run all the programs, and allow you to view the output. Help screens and message windows facilitate easy interaction with the software.

1.2.1.2

### PARSE

PARSE checks the syntax of the input file that contains the PLD design you created. If the program detects an error, it indicates where the error occurred. The error messages are sent to a file which you can retrieve. To correct as many errors as possible with each run, the program attempts to recover after each error. When the file is error-free, PARSE generates an intermediate file.

1.2.1.3

### EXPAND

EXPAND uses the intermediate file created by PARSE and performs the following functions:

- Expands the input equations
- Converts state machine syntax to Boolean equations

The software can only assemble Boolean equation fuse specifications. Therefore, if your input file contains a state machine design, EXPAND translates the design to Boolean equations.

The program expands XOR expressions to AND and OR expressions when the device does not contain an XOR gate. The XOR expressions will no longer be evident after running this program. (Also see the explanation and note on XORs in Section 1.2.1.4.)

EXPAND creates another intermediate file that contains expanded Boolean equations.

**Note**: You don't need to run EXPAND on a PROSE device.

1.2.1.4

### MINIMIZE

MINIMIZE uses the intermediate file created by PARSE or EXPAND to perform automatic logic reduction. This function enables you to use the space on your device more efficiently.

MINIMIZE looks for redundancy and minimizes AND and OR expressions. The program creates an intermediate file that is used by subsequent programs.

**Note**: When MINIMIZE detects an XOR gate on the device, the equations on either side of the XOR expression are minimized independently leaving the XOR intact. (See the note on XORs in Section 1.2.1.3.

**Note**: You don't need to run MINIMIZE on a PROSE device.

1.2.1.5

### XPLOT

XPLOT validates the architectural design of an input PAL design and produces fuse maps and JEDEC data. The program uses the intermediate file containing Boolean equations created by PARSE, EXPAND or MINIMIZE as input. XPLOT checks the equations for consistency and correctness for the specified device. When an error is detected, XPLOT attempts immediate recovery. In this way, XPLOT detects as many errors as possible on each run. Only if no errors are detected will the output fuse maps and JEDEC data be generated. XPLOT reads the architectural information for each device from files stored in the software containing profile descriptions for each device.

**Note**: XPLOT checks only valid Monolithic Memories PAL devices listed in Table 4-1.

1.2.1.6

### SIM

SIM checks the functionality of a PAL device design. You can run this program after XPLOT, or, if the design is architecturally correct, you can run SIM directly after checking the syntax with PARSE. SIM simulates the operation of the device you specify, calculating the output values based on input signals through the Boolean equations and any feedback.

SIM generates three output files: a history file, a trace file and a JEDEC test vector file. The history file shows the values of every pin through a simulation sequence. The trace file, which is a subset of the history file, shows only the pins you specify in the input file. If XPLOT has been run and a JEDEC fuseplot has been created, SIM adds test vectors to the JEDEC file that duplicate the simulation sequence when the device is tested on a programmer. All JEDEC checksums are recalculated.

**Note:** SIM tests only valid Monolithic Memories PAL devices listed in Table 4-1.

1.2.1.7

### PROASM-PROSIM

PROASM and PROSIM do for a PROSE PMS14R21 device what XPLOT and SIM do for PAL devices. The programs assemble and simulate PROSE device designs. PROASM accepts only state machine designs. Like XPLOT, the program generates a fuse map and a JEDEC file that can be downloaded to the programmer to program the PROSE device. Similarly, PROSIM generates history and trace files as well as JEDEC test data.

1.2.1.8

### PLSASM

PLSASM is the assembler for PLS devices and performs the same functions that XPLOT and PROASM do on PAL and PROSE devices.

**Note:** Currently, PALASM 2 software does not simulate PLS device designs.

1.2.1.9

### JEDMAN

JEDMAN disassembles JEDEC files and generates Boolean equations. The program allows you to read a fuse map directly from a programmed device.

1.2.1.10

### TREPL2

TREPL2 is a useful error detection tool that allows you to disassemble an intermediate TRE file and convert it to a Boolean equation input file. PALASM 2 software creates intermediate files after syntax checking, equation expansion, and minimization. TREPL2 can be used to disassemble any of these intermediate files. This means you can examine a Boolean equation input file:

- after the equations have been expanded
- after the equations have been minimized
- after the input state equations have been converted to Boolean equations

1.2.2

## Input, Output, And Intermediate Files

Table 4-3 lists the input, output, and intermediate files (files that the software creates but are not immediately visible to the user) required or generated by PALASM 2 software. Notice the extensions on each filename. It is important to use these extensions consistently since the software looks for them when retrieving files.

**Table 4-3**

**Input, Output, and Intermediate Files**

| Filename | Description |
|---|---|
| FILENAME.PDS | User defined PLD design input file |
| PALASM2.TRE | PLD intermediate design description |
| FILENAME.PDF | PLD architecture description data |
| FILENAME.LOG | Intermediate message file generated by PARSE, MINIMIZE, EXPAND, and PLSASM |
| FILENAME.XPT | PLD fuse map data |
| FILENAME.JED | PLD fuse JEDEC data |
| FILENAME.HST | Simulation history data |
| FILENAME.TRF | Simulation trace data |
| FILENAME.JDC | PLD fuse JEDEC data and JEDEC test vectors |
| FILENAME.PL2 | PDS file reconstructed from JEDEC output |
| FILENAME.JDM | JEDEC file that has been altered using JEDMAN as a supplementary program |

1.2.3

## PALASM 2 Supplementary Programs

The PALASM 2 software package currently includes several useful supplementary programs that are not supported by Monolithic Memories. An asterisk next to the program name in Table 4-4 indicates that the procedure to use it is described in the following chapters.

**Table 4-4**

**PALASM 2 Software Supplementary Programs**

| Program | Description |
|---|---|
| PDSCNVT | Conversion of previous PALASM version input file to PALASM 2 software syntax |
| *PC2 | Programmer communications program |
| *SCRSIM | Simulation waveform generation program |
| VTRACE | SIM output files to timing diagrams conversion |
| BINHEX | Binary to hexadecimal conversion |
| TIMING | Timing diagram entry program |
| PINOUT | Pinout program |
| DECODE | Address decoder program |

1.2.3.1

### PC2

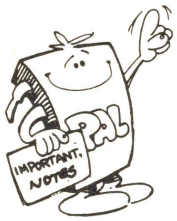
PC2 enables communication between PLD programmers and IBM-PC/XT/AT computers. PC2 is a menu-driven, multiple-choice program that guides you through various options for programming and checking PLDs.

1.2.3.2

## SCRSIM

SCRSIM takes the simulation output files, history and trace, and creates waveforms that can be viewed on the screen or sent to a printer.

Proceed to *Install PALASM 2 Software*, Chapter 2.


IMPORTANT NOTES

# 2. Install PALASM 2 Software

### *About This Chapter*

This chapter describes how to install the PALASM 2 software main menu and supplementary programs on IBM-PC/XT/ATs with either two floppy disk drives or a hard disk drive. It also describes how to add additional supplementary programs to the menu installation file.

The menu and supplementary programs are not available for VAX or UNIX systems. Monolithic Memories ships separate installation and operation instructions with the software for these systems.

| *To...* | *Refer to Section...* |
|---|---|
| Install the PALASM 2 software on a hard drive or with two floppy disk drives | 2.1 |
| Specify the editor and communications program | 2.2 |
| Add supplementary programs to the main menu | 2.3 |

2.1

## Installation Procedures

You can install the PALASM 2 software interactive menu and supplementary programs on IBM personal computers with either two floppy disk drives or with a hard disk drive. Before you start, make sure you have all the required equipment. Then read the section that describes your IBM configuration.

**Note:** The installation procedures update the AUTOEXEC.BAT and CONFIG.SYS files. Refer to *Modifications to AUTOEXEC.BAT and CONFIG.SYS Files*, Section 2.4, for more information.

2.1.1

## What You Require

To install PALASM 2 software, you require:

- An IBM-PC/XT/AT with either two floppy disk drives or a hard disk drive
- MS-DOS 2.1, or later versions
- Minimum memory of 384K bytes RAM and 2 megabytes on a hard disk
- PALASM 2 software on regular density disks for two floppy disk drives or high density disks for an IBM-AT
- Blank, formatted disks for making backup copies of the PALASM 2 software disk set

| *If your system uses...* | *Proceed to...* |
|---|---|
| Two floppy drives | Section 2.1.2 |
| A hard disk drive | Section 2.1.3 |

2.1.2

## Install PALASM 2 Software With Two Floppy Disk Drives

Before you load PALASM 2 software on an IBM-PC/XT/AT with two floppy disk drives, you should make a backup copy of the PALASM 2 software master disks. Refer to the IBM Disk Operating System reference manual for instructions. If any of the disks fail to load, contact your local Monolithic Memories representative.

If your system uses a hard disk drive, skip to *Install PALASM 2 Software On A Hard Disk Drive*, Section 2.1.3.

Follow these steps to load PALASM 2 software.

1. Insert the MS-DOS disk in drive B.
2. Insert the PALASM 2 software disk-1 in drive A.
3. To copy the MS-DOS system files to the PALASM disk, enter:
   - B: <return>
   - SYS A: <return>
   - Copy B: COMMAND.COM A:<return>
4. Remove the MS-DOS disk from drive B.
5. Re-boot the system by simultaneously pressing <ctrl> <alt> <del>.
6. Place the PALASM 2 software disk in drive B.
7. To start the PALASM 2 software program, enter:

   A:PALASM <return>.

   The screen displays the product and company name.
8. To display the main menu, enter <return> again. Figure 4-4 shows the main menu.
9. Proceed to *Software Setup*, Section 2.2, to define the editor and the communications program used to communicate with your programmer.

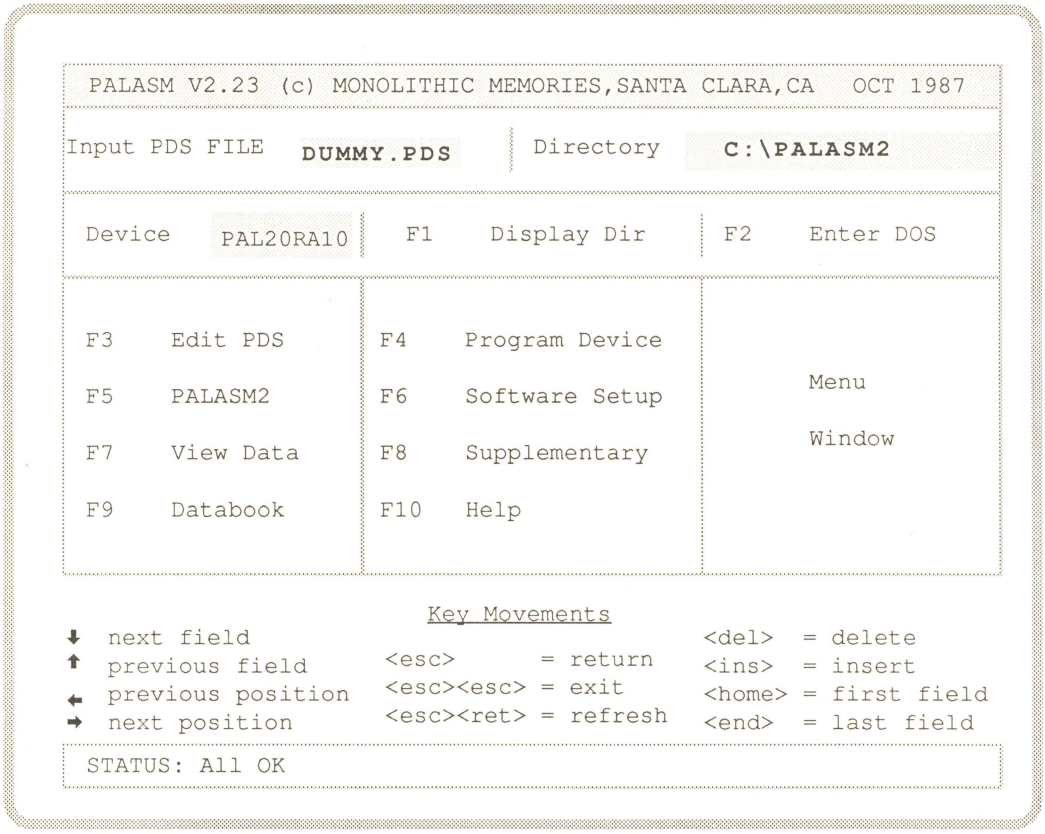



**Figure 4-4**

**PALASM 2 Software Main Menu**

2.1.3

## Install PALASM 2 Software On A Hard Disk Drive

When using an IBM-PC/XT /AT with a hard disk drive, install the main menu onto the specified drive. Then install the supplementary programs on the same drive. This section explains both installation procedures.

Before you install PALASM 2 software, you should make a backup copy of the PALASM 2 software master disks. If any of the disks fail to load, contact your local Monolithic Memories representative.

2.1.3.1

## Install The Menu

Follow these steps to install the main menu software on your IBM-PC/XT/AT hard disk.

1. At the system prompt, insert the PALASM 2 software master disk-1 in drive A.

2. To install the PALASM 2 software, enter:

   A:PAL2INST <return>

   The screen displays:

```
PALASM 2 Hard Disk Install Program
(c) CopyRight Monolithic Memories Inc. 1987

System Booted from Drive C

Which Drive to Install on? (default = C) _
```

3. Enter the name of the drive and press <return>. To select the default drive (C>), press <return>. The screen displays:

```
PALASM 2 Hard Disk Install Program
(c) CopyRight Monolithic Memories Inc. 1987

System Booted from Drive C

Which Drive to Install on? (default = C) _
... Installing to Drive = C  OK?  (Y/N) _
```

**Note:** The system boots from the logical drive you selected and the screen displays the letter designator for that drive.

| *If you enter...* | *Then...* |
|---|---|
| <N> <return> | Repeat step 3. |
| <Y> <return> | The screen displays: |

```
Making directory  ...  C:\palasm2
Making directory  ...  C:\palasm2\pal2
Making directory  ...  C:\palasm2\supl
Making directory  ...  C:\palasm2\pdf
Making directory  ...  C:\palasm2\msg
```

**Note:** If you are updating your PALASM 2 software, these files may already exist. If so, the screen will display a message that the software is making directories and that some of these directories already exist. When the screen prompt asks if you want to make directories with the same name, enter <Y> <return>.

The system updates the AUTOEXEC.BAT and CONFIG.SYS files and copies the files from drive A to the install hard drive. For a description of the changes to these files, refer to *Modifications To AUTOEXEC.BAT And CONFIG.SYS Files*, Section 2.4. When complete, the screen displays the input install request menu, as shown in Figure 4-5.

```
PALASM 2 Hard Disk Install Program
(c) CopyRight Monolithic Memories Inc.  1987


Input Install request


0 ...Exit Install Procedure
1 ...Install ALL PALASM 2 Software
2 ...Install Software for PAL devices only
3 ...Install Software for PROSE devices only
4 ...Install Software for PLS devices only
5 ...Install Supplementary Software only

?: _
```

Figure 4-5

PALASM Input Install Request Menu

4. You can now exit the install program by entering <0> <return>. To install the programs appropriate for your devices, proceed to *Load The Software*, Section 2.1.3.2.

2.1.3.2

## Load The Software

From the input install request menu, Figure 4-5, select the software installation option that applies to your application.

| *If you want to...* | *Then ..* |
|---|---|
| Install all the software programs at once, including the supplementary programs | • Select option 1.<br>• When installation is complete, exit the install program by selecting option 0.<br>• Proceed to *Verify The Installation*, Section 2.1.3.3. |
| Load selected software programs individually | Select options 2 to 5. The example below uses option 2 to install software for PAL devices only. |

Follow the steps in this example to install the software for PAL devices. To use this example for PROSE or PLS devices, make selections appropriate to the software you want to install.

1. Enter <2> <return>. The screen displays:

```
Insert PARSE.EXE in Drive A...Press any key to start_
```

2. Remove the PALASM 2 software master disk from drive A.

3. Insert the disk that contains the PARSE.EXE file into drive A. Each disk label lists the programs included on that disk.

4. Press any key. As the program copies each file from drive A to the install hard drive, the screen displays the name of that file. For example:

```
Copying A:PARSE.EXE to C:\palasm2\pal2\*.* ...
Copying A:PARSE.MSG to C:\palasm2\msg\*.* ...
```

After copying all the files on this disk, the screen prompts you to load a new disk which contains another file.

5. Repeat steps 3 and 4, as necessary, using the disks that contain the requested files. When the entire installation procedure is complete, the screen displays the input install request menu, as shown in Figure 4-5.

6. Now you can either install another program or exit the installation procedure:

   - To install another software program, repeat steps 1 through 5, making selections appropriate to the software you want to install.

     The installation software for the individual devices share some common programs. When you install a second program, you delete the common files from the hard disk and copy them again from the disk in drive A. You can avoid this duplication by selecting option 1, Install ALL PALASM 2 Software.

   - To exit the installation procedure, enter <0> <return>. The screen displays:

```
*** PALASM installation to hard disk completed ***


Now Remove Diskette from Drive A.


**********************************************
Re-Boot Machine then enter C:>PALASM
**********************************************

C>
```

Proceed to *Verify The Installation*, Section 2.1.3.3.

2.1.3.3

## Verify The Installation

After you have completed the installation procedure, follow these steps to run a test.

1. Remove the disk from drive A .

2. Re-boot your system (simultaneously press <ctrl> <alt> <del>).

3. At the system prompt, start PALASM 2 software by entering:

   C: PALASM <return>

   The screen displays the product and company name.

4. To display the main menu, enter <return> again. The screen displays the main menu as shown in Figure 4-4.

5. Proceed to *Software Setup*, Section 2.2, to define the editor and the communications program used to communicate with your programmer.

2.2

## Software Setup

After you have installed the PALASM 2 software, **you must tell the system the name of your text editor and programmer communications program**. To do this, use the main menu option F6, Software Setup.



1. From the main menu, press <F6>. The screen displays the installation menu, as shown in Figure 4-6.

2. Use the arrow keys to move the cursor to the Editor field.

   **Note:** P2EDIT is a fictitious name, intended to show the format for entering the name of your text editor.

3. Enter the name of your editor. Press the spacebar to delete all extra characters from the field.

   **CAUTION**

   The editor you use must generate clean ASCII text. Check your editor installation program to make sure the editor does not automatically load in a mode that embeds formatting control characters in your design files. For example, if you use Wordstar, make sure it automatically loads in non-document mode.

4. Move the cursor to the Programmer field.

   **Note:** pc2comm.com is a fictitious name, intended to show the format for entering the name of your programmer communication program

5. Enter the name of your programmer communication program. You can use a commercial communications program or use the PC2 program, which is included on the Supplementary disk.

6. To return to the main menu, press <esc>.

7. If you want to add supplementary programs to the menu, proceed to *Add Supplementary Programs To The Menu*, Section 2.3. If not, skip to *Modifications to AUTOEXEC.BAT And CONFIG.SYS Files*, Section 2.4.

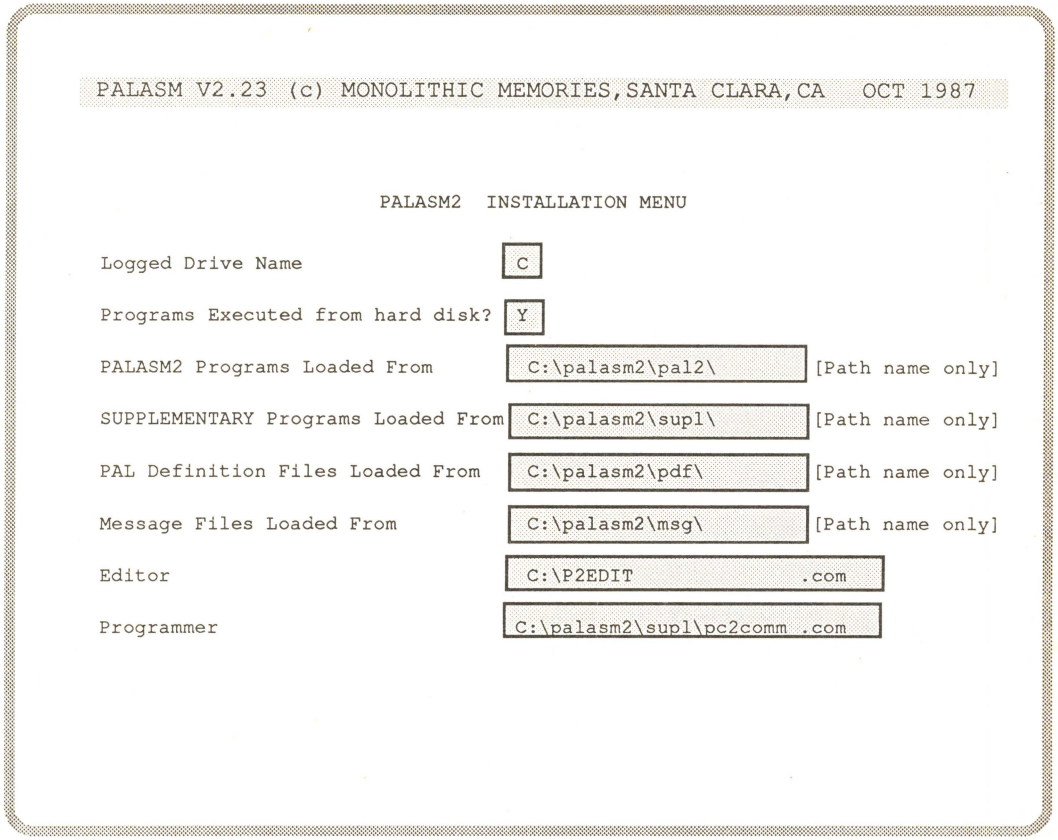



**Figure 4-6**

**PALASM 2 Software Installation Menu**

2.3

## Add Supplementary Programs To The Menu

After installing the supplementary programs on your hard disk, you can add these programs to the installation procedure stored in the MENU.SYS file. Then you can call a supplementary program from the main menu by selecting option F8.

Table 4-5 lists the supplementary programs you can add to the MENU.SYS system file.

Table 4-5

Additional Supplementary Programs

| Program | Function |
|---|---|
| SCRSIM.COM | Simulation waveform generation program that is called automatically when you use option F7, View Data |
| VTRACE.COM | A utility program to print simulation output files as timing diagrams |
| BINHEX.COM | A binary-to-hexadecimal conversion program |
| TIMING.COM | Timing diagram entry program |
| PINOUT.COM | Generates a list of the pin names from the .TRE file (created by the PALASM 2 software assembler) |
| DECODE.COM | Address decoder program that generates PALASM 2 software Boolean equations |

To add a program to the MENU.SYS file, follow these steps.

1. Enter MS-DOS from the PALASM 2 software main menu by pressing <F2>.

2. Enter the text editor. Refer to the text editor operation manual for instructions.

3. Open the MENU.SYS file.

4. To the end of the file, add the filename of the supplementary program in the following format. Any change you make to the file must occur after the Input file name (see Figure 4-7).

   $<FILENAME>

   For example, to add the program PINOUT.COM, enter:

   $PINOUT.COM

5. Save the file.

6. To return to the PALASM menu, press <esc>. The screen displays the main menu.

7. To view a list of the supplementary programs, press <F8>. Make sure the programs you added are included in the list.

Refer to Figure 4-7 for the MENU.SYS file format.

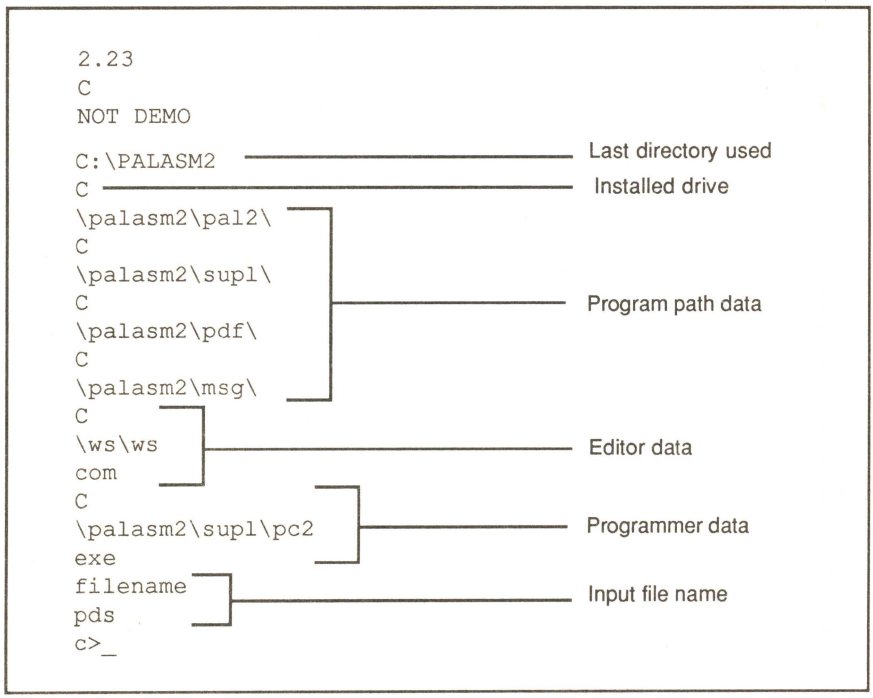



Figure 4-7

MENU.SYS

2.4

## Modifications To The AUTOEXEC.BAT And CONFIG.SYS Files

The installation process modifies your AUTOEXEC.BAT and CONFIG.SYS files. If you did not have these files, they are created during the installation. The following lines are added to the AUTOEXEC.BAT file:

```
REM C PALASM 2 path statement
PATH C:\;C:\palasm2\supl;%path%
ECHO Palasm 2 Software Installed - (c) Copyright MMI 1987
All Rights Reserved
```

The following line is added to your CONFIG.SYS file

Files=20

If you already have a CONFIG.SYS file with a Files = attribute greater than 20 (for example, Files = 30), then the CONFIG.SYS will not be modified.

Proceed to *Run the Software*, Chapter 3.

# 3. Run the Software

### *About This Chapter*

This chapter demonstrates how to run the software with an example input file – SUPER.PDS. This input file contains a Boolean equation design for the PAL16R6.

The procedure steps you through PALASM 2 software by selecting options from the menu. When the software processes are complete, you view the output files on your screen.

| *For a description of...* | *Refer to Section...* |
|---|---|
| Running the software through the menu | 3.1 |
| Autorun assembly and simulation | 3.5 |
| Processing an input file | 3.6 |
| Viewing the assembly output files | 3.6.5 |
| Simulating an input file | 3.7 |
| Viewing the simulation output | 3.7.2 |
| Interpreting the assembly output files | 3.10 |

If you want to run the software from DOS instead of using the PALASM 2 software menu, skip to *Run The Software From DOS*, Section 3.11.

3.1

## Overview Of The Procedure

The flowchart in Figure 4-8 shows the procedure to run PALASM 2 software. Figure 4-9 shows the basic processing options in a flowchart. The steps are as follows.

1. Check the syntax of the input file.
2. Expand the input equations.
3. Minimize the input equations.
4. Assemble the file and generate JEDEC output.
5. Simulate the design.

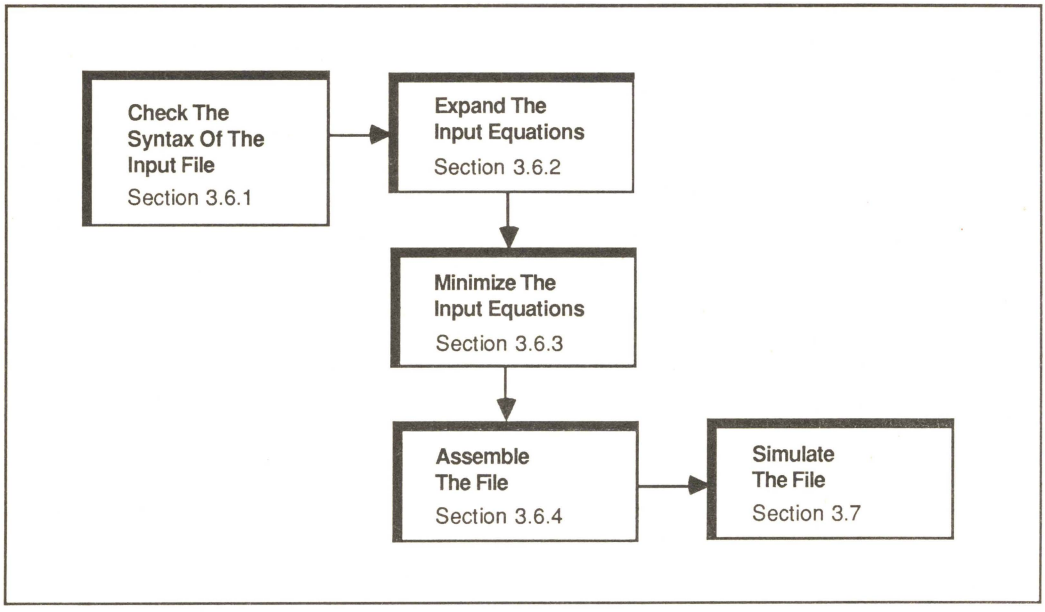



Figure 4-8

PALASM 2 Software Processing Sequence

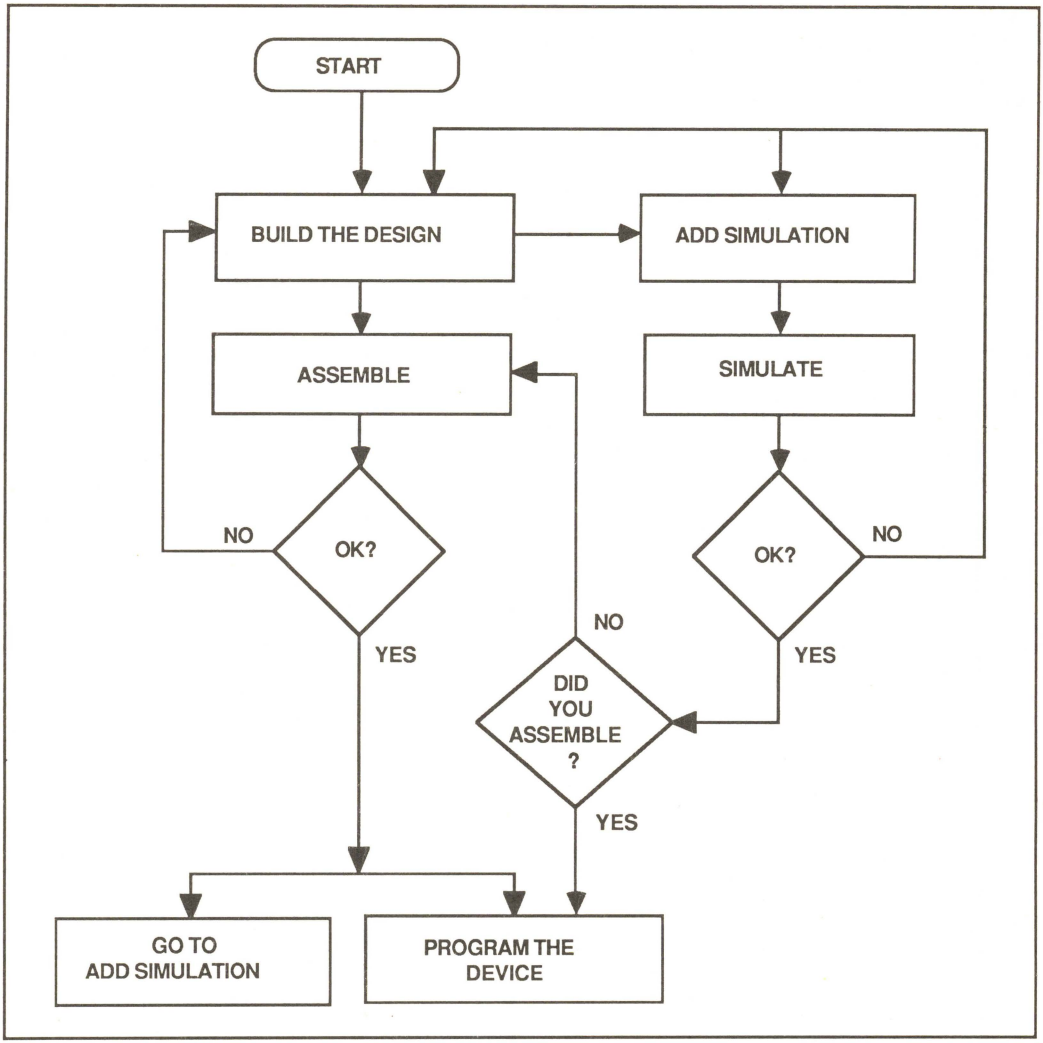



Figure 4-9

Basic Design Options

Before you run the programs to accomplish these steps, you must open the PALASM menu and become familiar with its operation, as explained in the next section.

3.2

## Prepare To Use The Menu

The following list shows what tasks you should complete before using the menu.

1. Install the menu on your computer following the procedure in *Install The Menu*, Section 2.1.3.1.

2. Install the text editor of your choice on the default drive. Use option F6, Software Setup, on the main menu. Refer to *Software Setup*, Section 2.2, for the procedure.

3. Install the programmer communication software on your default drive. This too is done on the software setup menu. Refer to *Software Setup*, section 2.2 for the procedure.

**Note**: If you are using a twin floppy system, keep all the PALASM 2 software disks by your computer. The PALASM menu prompts you to insert disks in drive B when it needs a particular program. The disks are identified by the program name on the label.

Now you are ready to use the PALASM menu.

3.2.1

## Call The PALASM Menu

To call the PALASM menu, type

PALASM

The first screen that you see displays the product and company name. Press <return> to display the main menu as shown in Figure 4-10.

The main menu contains four features that you should identify.

1. Fields

   You enter data in these fields.

2. Function keys

   You use these keys to open sub-menus or activate a program.

3. Key movements

   The key movements, displayed at the bottom of your screen, allow you to:

   - Move the cursor between the fields by using the arrow keys or <home> and <end> on your keyboard
   - Return to the main menu from another menu, by pressing <esc> on your keyboard
   - Refresh the screen by pressing <esc> <return> on your keyboard.
   - Exit the program by pressing <esc> <esc> on your keyboard.

4. Status line

   This is a reserved area at the bottom of every menu screen. Read the messages displayed on the status line whenever they change.

Identify the four features on your screen with the help of Figure 4-10. As you step through the procedure for running the software, you will see how to use these features.

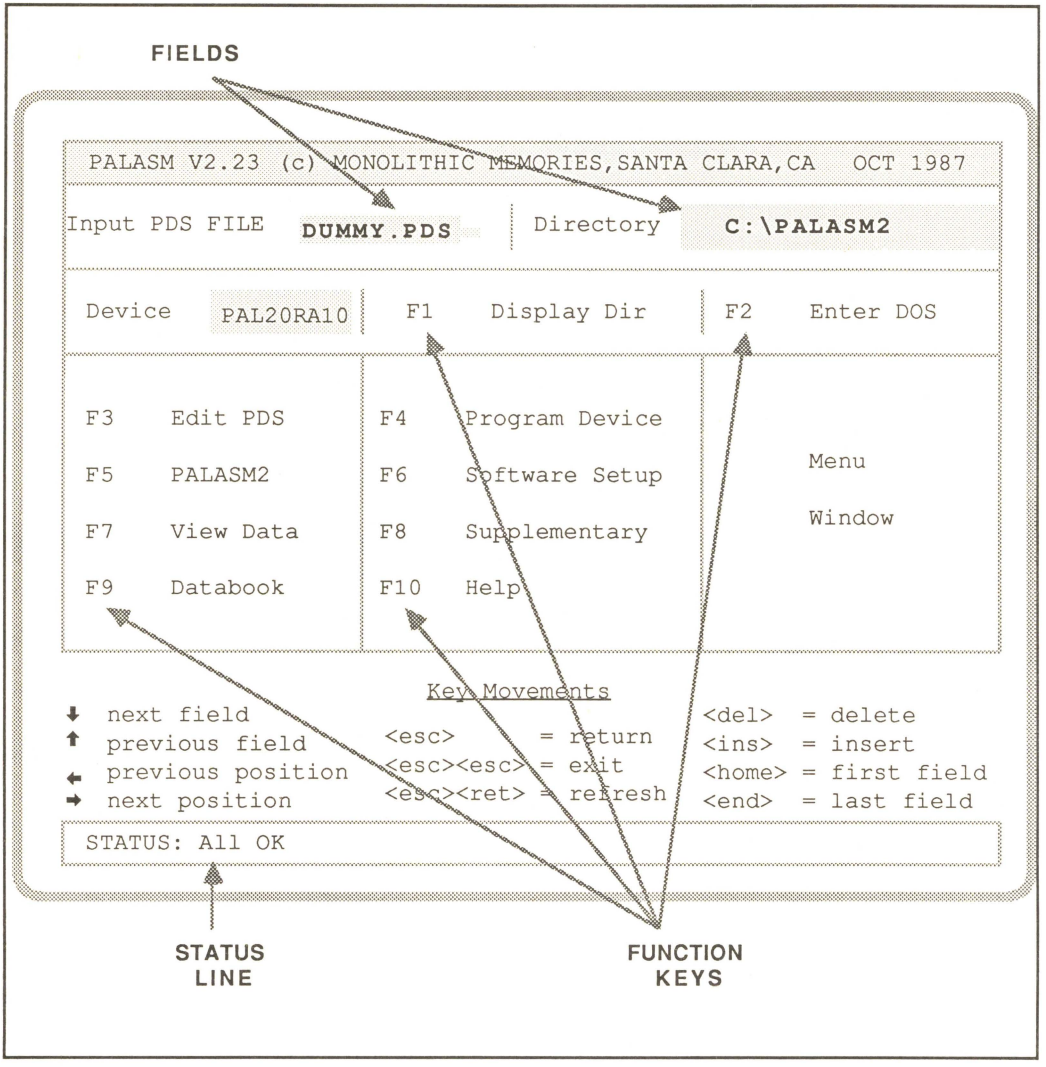



Figure 4-10

The Main Menu

3.2.2

## Specify The Directory And Input File

The following fields appear at the top of the menu.

- Input PDS File
- Directory

**Note**: You cannot enter the device name on the menu. It appears automatically when the software indentifies the input file.

Notice that Figure 4-10 displays the default drive, C, in the directory field. Also, the dummy filename appears in the input PDS file field. The example file you need is SUPER.PDS. The PALASM menu looks for the input PDS file in the default directory displayed on the menu. To make sure that the input file SUPER.PDS is on the default drive, follow these steps.

**Note**: If you have a twin floppy system, your default drive should be drive B. Insert the Design Examples disk in drive B and follow steps 1 to 5.

1. To display the files on the default directory, press <F1>.

   F1 Display Dir

   The menu screen displays the files on the default directory. Make sure that SUPER.PDS is in the directory. If not, go to step 2. If SUPER.PDS is in the directory, press any key to return to the main menu and go to step 5.

2. To change the default directory, press <F2>.

   F2 Enter DOS

   You are now in DOS. List your directories and find the file SUPER.PDS. Make a note of the directory it is in.

3. Press any key to return to the main menu.

4. On the main menu, enter the correct directory name in the directory field.

5. Next, enter SUPER.PDS in the input PDS file field.

   Input PDS file `SUPER.PDS`

3.3

## Open The Sample Input File

The file SUPER.PDS is now displayed in the data entry field at the top left of your screen.

To open the file in your editor, press <F3>.
F3 Edit PDS

The editor you specified on the software setup menu (see *Software Setup*, Section 2.2) now displays SUPER.PDS.

**Note**: While you are in the editor, your editor commands apply. When you exit the editor using the editor command, you automatically return to the PALASM main menu.

3.4

## Study The Sample Input File

If you are not familiar with PALASM input files, take a moment to study the sample file. Otherwise exit the editor and, go to *Autorun Assembly And Simulation*, Section 3.5.

Scroll through the editor and glance at the various segments in the PDS file. Each segment begins with a header, such as EQUATIONS, or SIMULATION. Chapters 4 and 5 of this manual describe how to create an input file. Chapter 6 of this manual describes how to create the simulation segment. When you create an input file, you use the editor specified in the software setup menu. For now, the sample input file will demonstrate the menu operation.

**CAUTION**

Do not make any changes to the sample file SUPER.PDS since it is error-free and ready for processing.

To return to the main menu, exit the editor.

3.5

## Autorun Assembly And Simulation

The PALASM menu program offers a time-saving autorun feature that allows you to run the assembly and simulation programs with one keystroke. Autorun is fast and easy, but to become familiar with individual processes you should run each process separately. Also, if your input file contains errors, it is easier to find out exactly where the error is if you run the processes individually.

For now, you may either use autorun or run each of the programs individually.

Proceed to the next section to find out about running the programs individually. The procedure to use the autorun feature follows.

1. Select the PALASM2 option by pressing <F5>.

   F5 PALASM2

   The menu window displays the PALASM 2 sub-menu.

2. Select the Autorun 1-5 option by pressing <6>.

   6 Autorun 1-5

   The lower window opens and processing begins.

3. Watch the status line carefully as the following operations are completed.

   - syntax check
   - expansion
   - minimization
   - assembly
   - simulation

4. When you see the message

   ```
   SIM  File Processed Successfully
   ```

   press <esc>.

The assembly and simulation processes generate output files which you can view. One of these files is the JEDEC file which is required by the device programmer.

| *If you want to...* | *Refer to...* |
|---|---|
| Run each assembly and simulation step separately | Section 3.6 |
| View the assembly output | Section 3.6.5 |
| View the simulation output | Section 3.7.2 |

The options on the PALASM2 sub-menu allow you to run steps 1-6 on the main menu individually. The procedure that enables running each program individually follows.

3.6

## Process The Input File

This section describes the procedure to process the input file by running each assembly and simulation step separately. After completing this procedure, you will have a JEDEC file that enables you to program a device. The steps involved in processing the input file are listed below and explained in detail in the following pages.

1. Check the syntax of the design file
2. Expand the input equations
3. Minimize the input equations
4. Assemble the file

A description of these steps follows.

3.6.1

## Check The Syntax Of The Input File

PALASM 2 software checks the syntax of the input PDS file. The software displays messages that identify errors. These messages are put into a file that you can either view on your screen or send to a printer.

Since you are now using a sample input file which contains no errors, the syntax check operation should be successful. The procedure to check the syntax of the input file follows.

1. Make sure to specify the correct input file and directory. Refer to *Specify The Directory And Input File*, Section 3.2.2, for instructions.
2. Press <F5> to access the PALASM2 sub-menu.

   F5 PALASM2

   The menu window displays a sub-menu as shown in Figure 4-11.

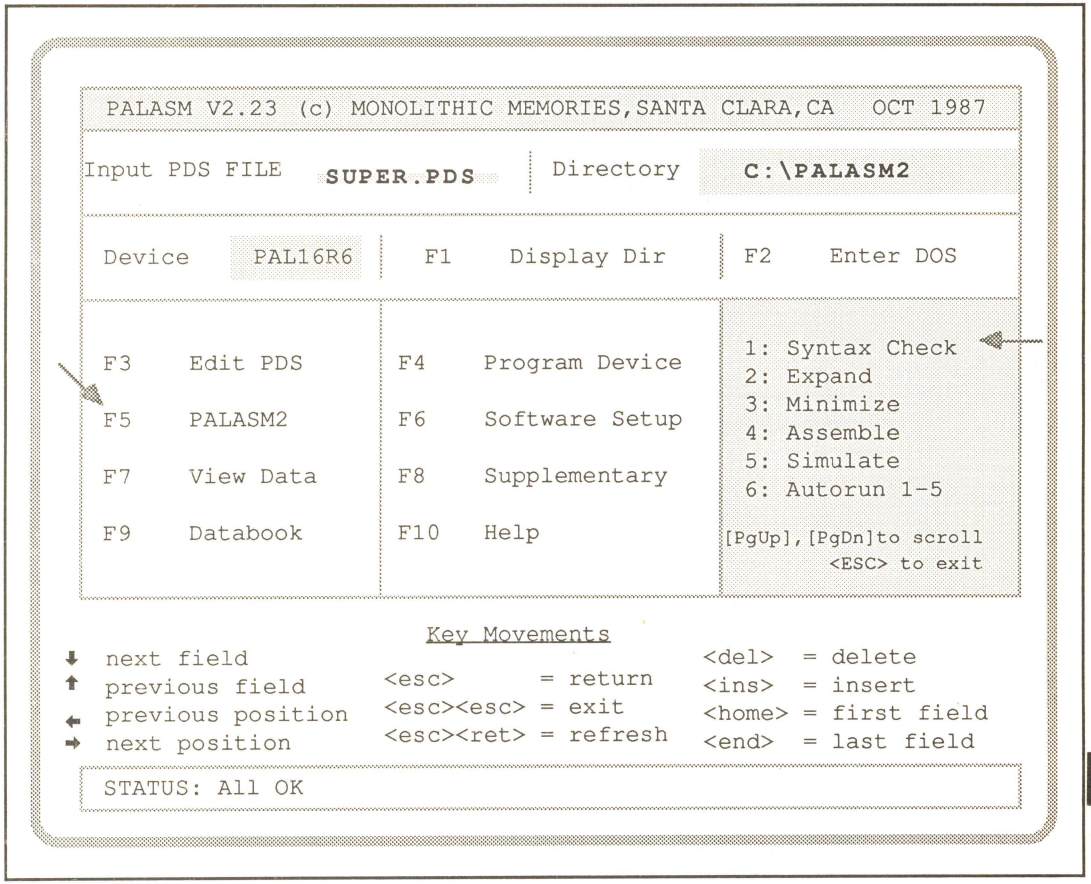



**Figure 4-11**

**Page One of the PALASM2 Sub-menu**

3. The sub-menu has six options on page one and two options on page two. Use <PgUp> or <PgDn> to view the two additional options. To check the syntax of the input file SUPER.PDS, select the syntax check option on page one by pressing <1>.

   1 Syntax Check

   The lower window opens in the lower part of the main menu. You can view the syntax check operation, or look at the message on the status line at the bottom of your screen. When the syntax check operation is complete, the screen displays the message shown in Figure 4-12.

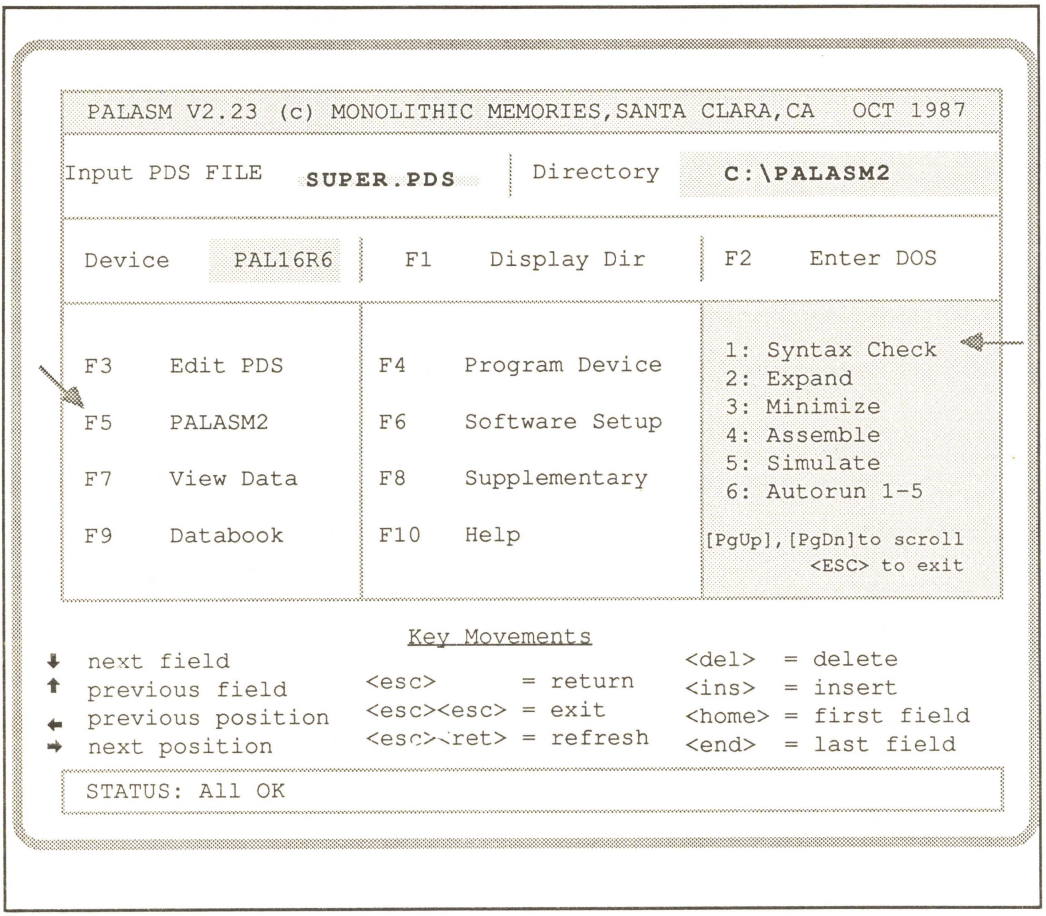



Figure 4-12

The Syntax Check Operation

### 3.6.2

## Expand The Input Equations

PALASM 2 software expands the input Boolean equations before minimizing them. If your input file is in state machine syntax, the equations are converted to Boolean equations at this stage.

**Note**: Skip this step if you are using the PMS14R21 PROSE device. Expansion is performed automatically on a PMS14R21 input file.

The procedure to expand the input equations follows.

1. Select the Expand option on the PALASM2 sub-menu by pressing <2>.

   2 Expand

   The lower window now displays the expansion process.

2. When the process is complete, the status line displays the following message

   ```
   Expand Process Successful.
   ```

Proceed to *Minimize The Input Equations*, Section 3.6.3.

3.6.3

## Minimize The Input Equations

After expanding the input equations, the next step is to minimize the equations. This step minimizes Boolean equations. Minimization also ensures that the space on the device is used efficiently.

**Note**: Skip this step if you are using the PMS14R21 PROSE device. Minimization is performed automatically on a PMS14R21 input file.

The procedure for minimizing input equations follows.

1. Select option <3> on the PALASM2 sub-menu.

   3 Minimize

   The lower window now displays the minimization operation and the status line displays messages.

2. As soon as minimization is complete, the lower window displays the following message

   ```
   Minimize Program Successful
   ```

Proceed to *Assemble The Input File*, Section 3.6.4.

3.6.4

## Assemble The Input File

The assembly program generates the JEDEC file which is required to program the device. The procedure to run the assembly program follows.

1. Select option <4> on the PALASM2 sub-menu.

   4 Assemble

   The lower window now displays the assembly process and the status line displays messages.

2. As soon as the input file has been assembled, the lower window displays the following message

   ```
   The fuseplot is stored in SUPER.XPT.
   The JEDEC  is stored in SUPER.JED.
   ```

   Note the names of the output files.

3. Press <esc> to exit the sub-menu.

The output file SUPER.JED is the JEDEC file required by the programmer to program the device. *Program The Device*, Chapter 7, describes the programming procedure. Proceed either to Chapter 7 or proceed to *View The Assembly Output Files*, Section 3.6.5.

3.6.5

## View The Assembly Output Files

You can view the output files that the PALASM 2 software assembler generates or proceed to *Simulate The Sample Design*, Section 3.7.

Figure 4-13 shows the output files that the assembler and simulator generate.

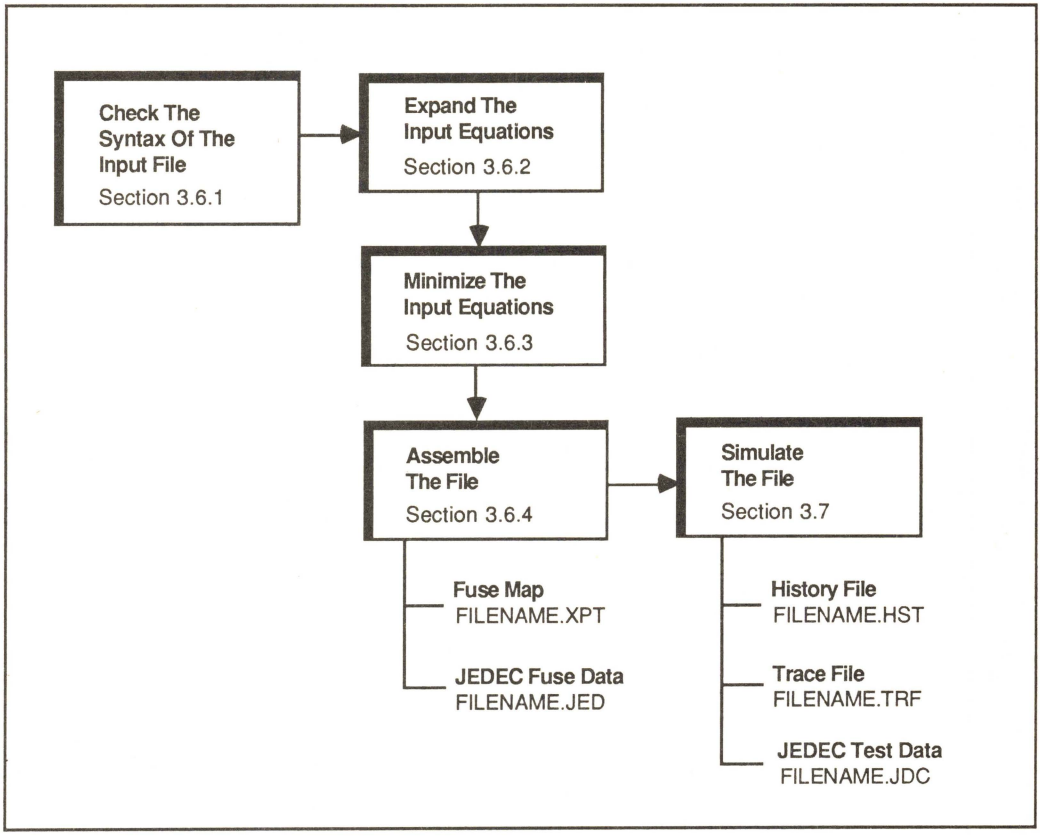



**Figure 4-13**

**Assembly and Simulation Output Files**

The PALASM 2 software assembler generates two output files. They are:

- The fuse map — SUPER.XPT
- The JEDEC fuse data — SUPER.JED This file is required by the programmer to program the device.

The procedure to view these files on your screen follows.

1. Select the View Data option on the main menu by pressing <F7>.

   F7 View Data

   The menu window now displays page one of the View Data sub-menu.

2. Notice the message at the bottom of the View Data sub-menu. To display page two of the View Data sub-menu, press <PgDn> on your keyboard.

   <Pg Dn>

   The menu window displays page two of the View Data sub-menu as shown in Figure 4-14.

2. Select the Fuse Map option by pressing <1>.

   1 Fuse Map

   The entire screen now displays the fuse map.

3. To return to the main menu, press <esc>.

4. Select the JEDEC Fuse Data option by pressing <2>.

   2 JEDEC Fuse Data

   The entire screen now displays the JEDEC fuse data. Press any key to scroll down and <esc> to return to the main menu.

5. Press <esc> on your keyboard to close the sub-menu.

To read the output files, refer to *Interpret The Assembly Output Files*, Section 3.10.

This completes the assembly procedure. To simulate the sample input file, proceed to the next section. To create your own design, proceed to *Create A Boolean Equation Design*, Chapter 4, or *Create A State Machine Design*, Chapter 5.

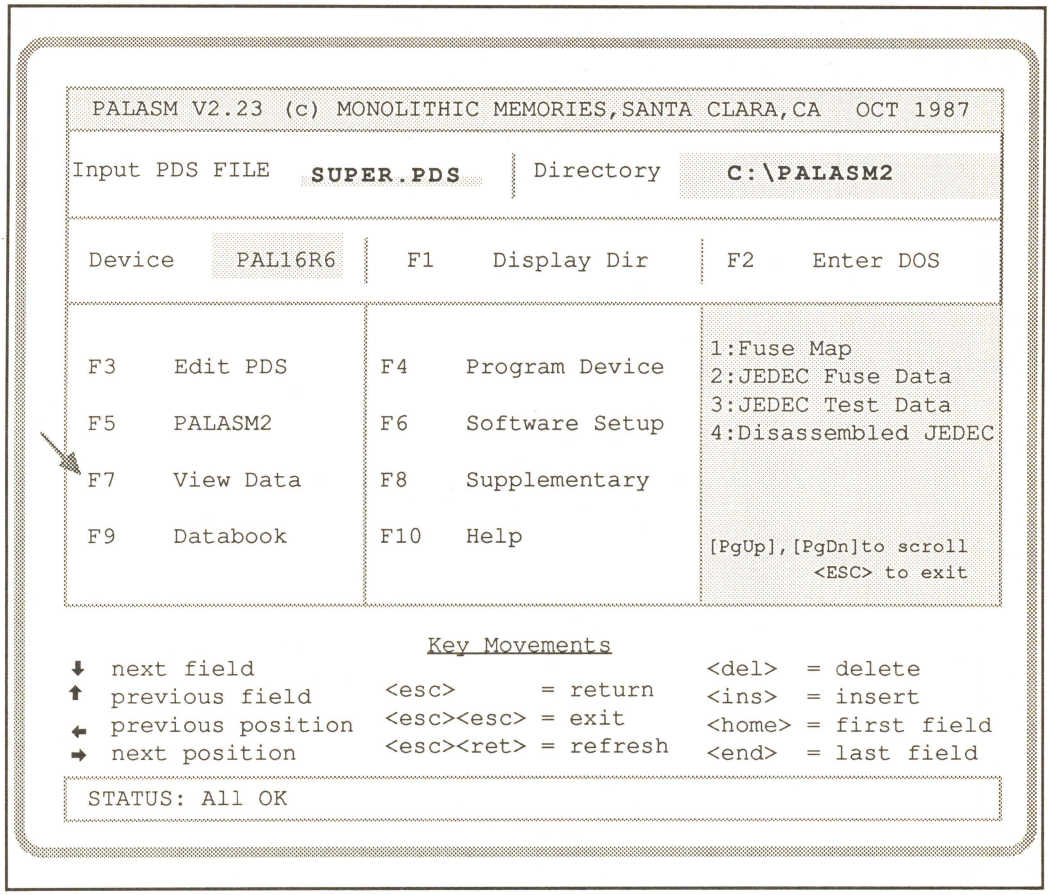



**Figure 4-14**

**Page Two of the View Data Sub-menu**

3.7

## Simulate The Sample Design

Simulation allows you to test your design without actually programming a device. The sample input file SUPER.PDS contains a simulation segment. You cannot simulate your design without including this segment in your input file. *Build Simulation*, Chapter 6, describes how to set up the simulation segment in the input file.

This section describes how to run the simulation program and view the output on your screen.

3.7.1

## Run The Simulation Program

1. If the PALASM2 sub-menu is not displayed in the menu window, press <F5>.

   F5 PALASM2

   The sub-menu now displays page one of the PALASM2 sub-menu as shown in Figure 4-11.

2. To select the Simulate option, press <5>.

   5 Simulate

   The lower window now displays the simulation operation and the status line displays messages.

3. Once simulation is complete, the lower window displays the following message

   ```
   Sim File Processed Successfully
   ```

4. Press <esc> to return to the main menu.

To check whether the design performed as planned, you must look at the simulation output.

3.7.2

## View The Simulation Output Files

The four ways in which to view the simulation output are:

- History File
- History Waveforms
- Trace File
- Trace Waveforms

The simulation process generates history and trace files from which the waveform program takes its information.

If you simulate your design after assembling it, the simulation program creates a JEDEC test data file which can be used for functional testing on the device programmer. The test vectors are added to the JEDEC fuse map that the assembler creates.

The simulation output is best viewed as waveform on your screen. To view the simulation output files, select the View Data option by pressing <F7>.

F7 View Data

The menu window displays page one of the View Data sub-menu as shown in Figure 4-15.

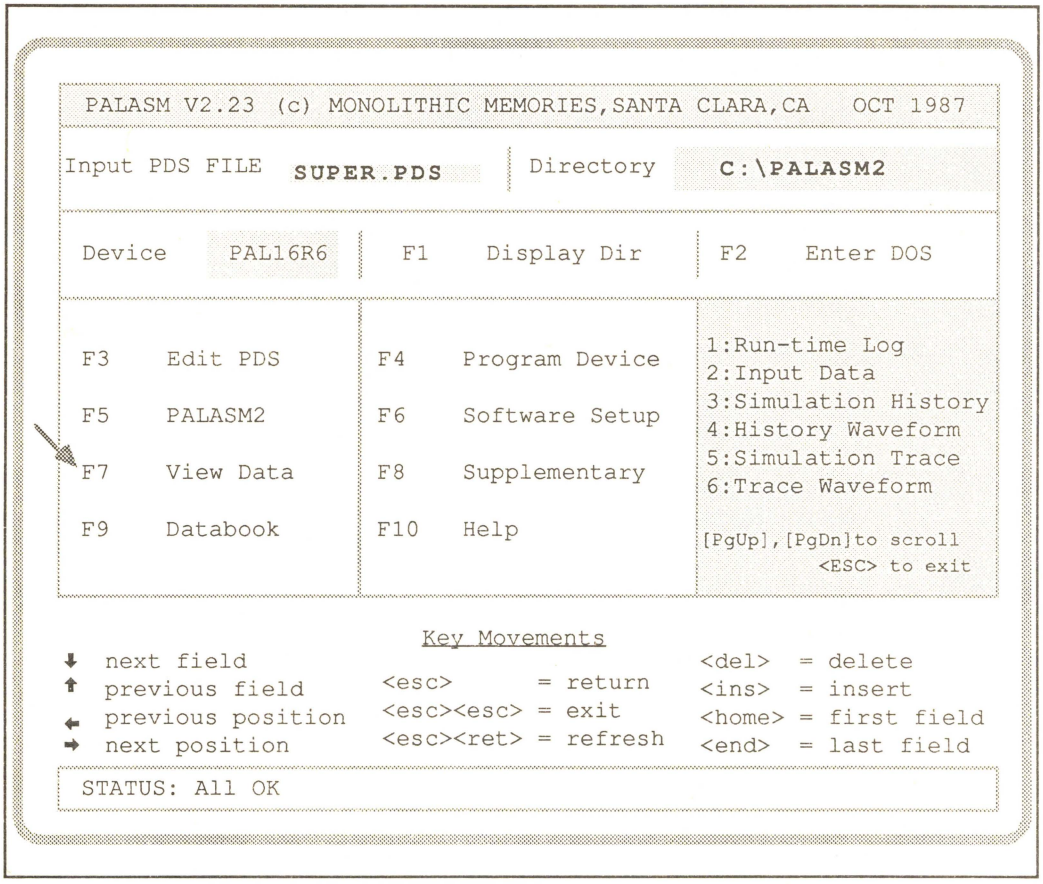



Figure 4-15

Page One of the View Data Sub-menu

To display the history or trace waveforms or the history and trace files follow these steps.

1. While the screen displays the View Data sub-menu, select the appropriate number on your keyboard.

   | | |
   |---|---|
   | 4 | History Waveform |
   | 6 | Trace Waveform |
   | 3 | Simulation History |
   | 5 | Simulation Trace |

The screen displays the output file. Figure 4-16 displays history waveforms.

2. Use the vertical bar cursor to track the values by selecting <B>.

3. Press <esc> to return to the main menu.

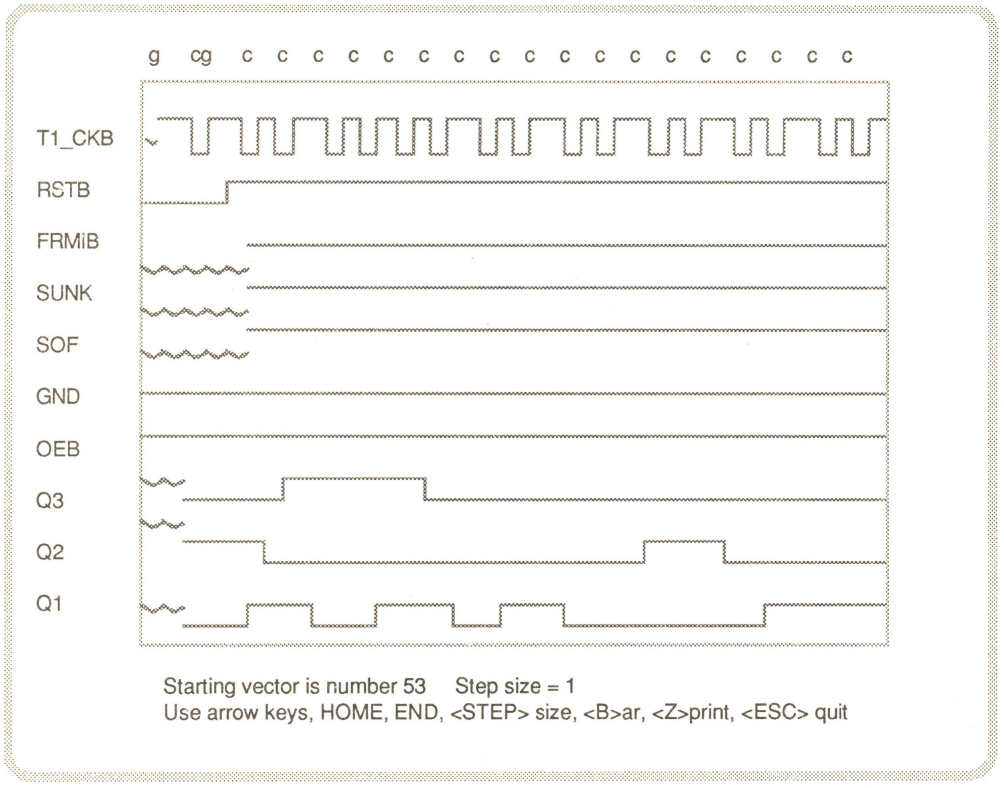



**Figure 4-16**

**History Waveforms**

3.7.3

## View The JEDEC Test Data

The simulation program creates a new JEDEC file by adding test data to the fuse map created by the assembler. This occurs only if simulation follows assembly . The JEDEC file created by the simulator contains test vectors and can be used to test and verify the device on the programmer.

To view the JEDEC test data file, follow these steps.

1. Select the View Data option on the main menu by pressing <F7>.

   F7 View Data

   The menu window displays page one of the View Data sub-menu.

2. Scroll down to page two by pressing <PgDn>. The screen displays page two as shown in Figure 4-14.

3. Select the JEDEC Test Data option by pressing <3>.

   3 JEDEC Test Data

   The entire screen displays the JEDEC test data file. You can scroll up and down by using <PgUp>, <PgDn>, and the arrow keys. Notice that the file contains both fuse and test data.

4. Press <esc> to return to the main menu.

For information on how to interpret the simulation output files, turn to *Review A Sample Input File And Interpret The Output Files*, Section 6-3.

3.8

## Disassemble A JEDEC File

The PALASM menu program offers an additional processing option that you do not always require. If you want information on the basic processing procedure alone, you may skip this section.

This process allows you to disassemble an existing JEDEC file and convert it to a Boolean equation input file.

The procedure to run the JEDEC disassembly program follows.

1. Select the PALASM2 option on the main menu by pressing <F5>.

   F5 PALASM2

   Page one appears in the menu window.

2. Scroll to page two by pressing <PgDn>. Page two appears in the menu window as shown in Figure 4-17.

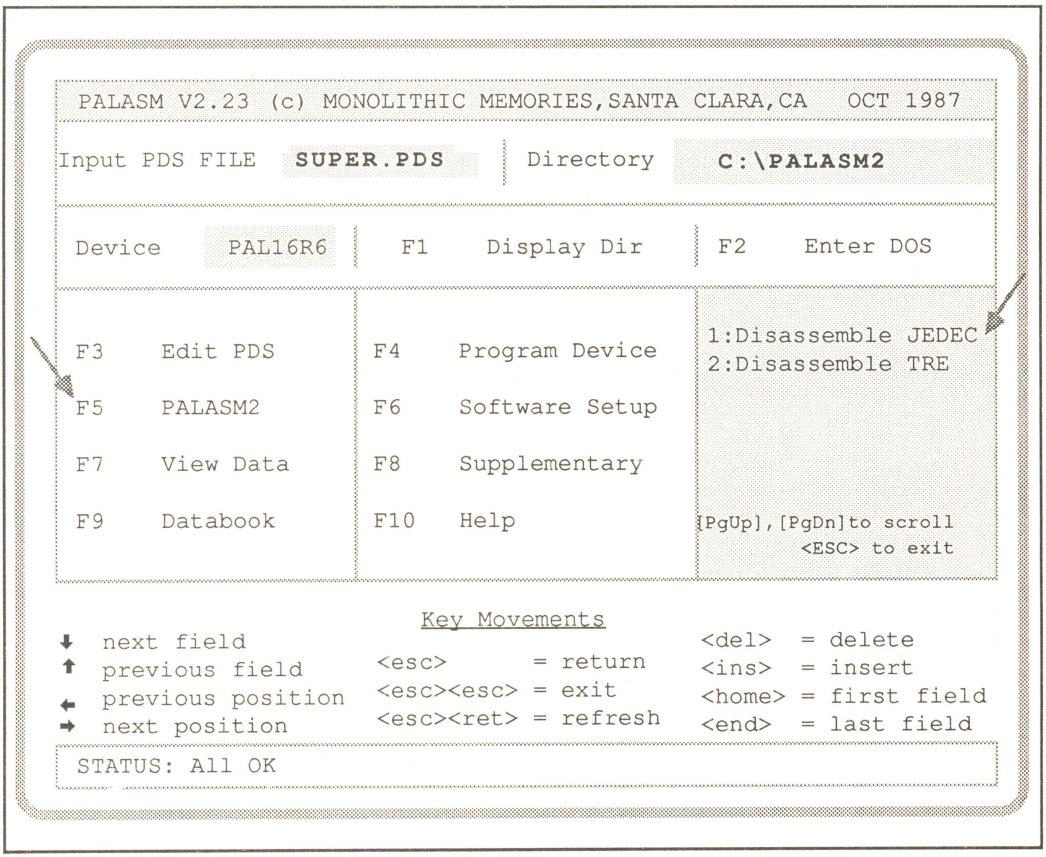



**Figure 4-17**

**Page Two of the PALASM2 Sub-menu**

3. Select Disassemble JEDEC by pressing <1>.

   1 Disassemble JEDEC

4. The JEDEC file produced by the assembler is now disassembled. The lower window displays the process. The status line displays the message

   ```
   Disassembling JEDEC file <SUPER.JED>
   ```

5. When the disassembly process is complete, the lower window displays the message

   ```
   %%JEDMAN%% Program successfully completed.  Check output
   files.
   ```

   The status line displays the message

   ```
   All OK.
   ```

6. Press <esc> to close the menu windows.

To view the disassembled JEDEC file, select the View Data sub-menu. Scroll to page two as shown in Figure 4-14 by pressing <PgDn> or <PgUp>. Select the Disassembled JEDEC option by pressing <4> on your keyboard.

3.9

## Identify Errors In The Input File

PALASM 2 software creates the following files that are useful error detection tools:

- Run-time Log
- Intermediate TRE files

You can identify the design errors in your input file by viewing these files.

3.9.1

## View The Run-Time Log

The run-time log contains any errors that the processes discover. The messages are sent to a file which you can either view on screen or print.

It is important to remember that the run-time log after each process is overwritten by the log of the succeeding process. This means that the log created after the syntax check process is overwritten by the log created after the expansion process. Also, the log files are deleted when you exit the menu. If you want to maintain each log, you can print them at the end of each process. The log files are all called MENU.LOG. (Also see the note on saving the file at the end of this section.)

**Note**: If you use the autorun feature, the log file records messages from all the processes.

To view the run-time log, follow these steps.

1. Select the View Data option on the main menu by pressing <F7>.

   F7 View Data

   The menu window displays the View Data sub-menu.

2. Select the Run-time Log option by pressing <1>.

   1 Run-time Log

   The entire screen now displays the run-time log for the last process you ran. You can scroll down by pressing any key.

3. To print the log, press <P>.

4. To return to the main menu, press <esc>.

**Note**: To save the file, select the DOS option on the main menu and change the filename of the file MENU.LOG.

3.9.2

## Disassemble The TRE File

PALASM 2 software creates an intermediate TRE file at the end of each of the following processes:

- syntax checking
- expansion of input equations
- minimization of input equations

Figure 4-18 shows when you can disassemble the TRE file.

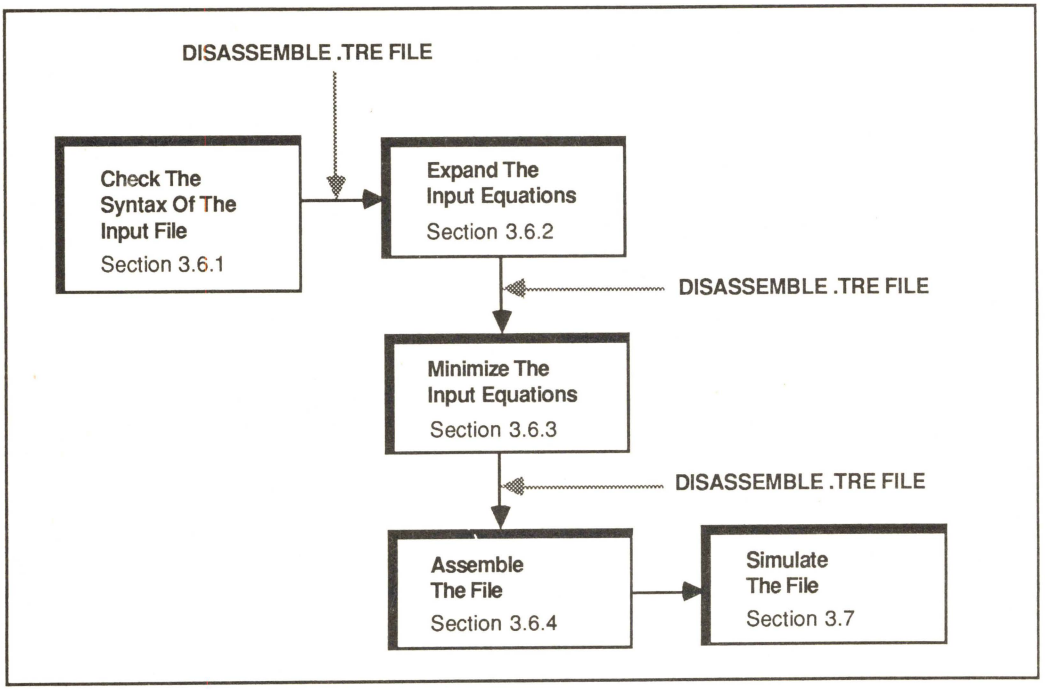



Figure 4-18

Disassemble The TRE File

The software uses the TRE file to perform processing functions. However, you are given the option of converting the TRE file to a Boolean equation input file. This is particularly useful in the following instances:

- When your input file was originally in state equations, TRE file disassembly performed after expansion gives you a Boolean equation input file.
- After minimization, TRE file disassembly gives you an input file with the equations minimized.

To disassemble the TRE file follow these steps *after the relevant process.*

1. Select the PALASM2 option on the main menu by pressing <F5>.

   F5 PALASM2

   The menu window displays page one of the PALASM2 sub-menu.

2. Scroll down to page two by pressing <PgDn>. The menu window displays page two.

3. Select Disassemble TRE by pressing <2>.

   2 Disassemble TRE

4. The lower window displays the TRE disassembly process. The status line displays the message

   ```
   Disassembling file - PALASM2.TRE
   ```

   When the process is complete the status line message reads

   ```
   All OK
   ```

5. Press <esc> to return to the main menu.

The TRE file is stored as run-time log. The procedure to view the TRE file follows.

1. Select the View Data option by pressing <F7>.

   F7 View Data

   The menu window displays the View Data sub-menu.

2. Select Run-time Log by pressing <1>.

   1 Run-time Log

The entire screen displays the disassembled TRE file.

3. To scroll down press any key.

4. You can print the file by pressing <P>.

5. To return to the main menu, press <esc>.

**Note**: When you exit the menu, the file will be deleted. To save the TRE file, select the DOS option on the main menu and change the filename of the file MENU.LOG.

3.10

## Interpret The Assembly Output Files

The assembler creates the following output files:

- the fuse map SUPER.XPT
- the JEDEC SUPER.JED

The files always have the extension .XPT or .JED as shown above.

The fuse map displays the programmed and unprogrammed fuses that the input file specifies. If you plan to simulate your design, you need not examine this file. The JEDEC is read by the device programmer and contains information required to program the device.

3.10.1

## Interpret The Fuseplot

Figure 4-19 displays a sample fuseplot. Notice the use of the following symbols.

Unprogrammed fuse x
Programmed fuse –

| *If you want to...* | *Then...* |
|---|---|
| View the JEDEC file | Proceed to Section 3.10.2 |
| Run PALASM 2 software from DOS, instead of the menu | Skip to Section 3.11 |
| Build a Boolean equation design | Skip to Chapter 4 |
| Build a state machine design | Skip to Chapter 5 |

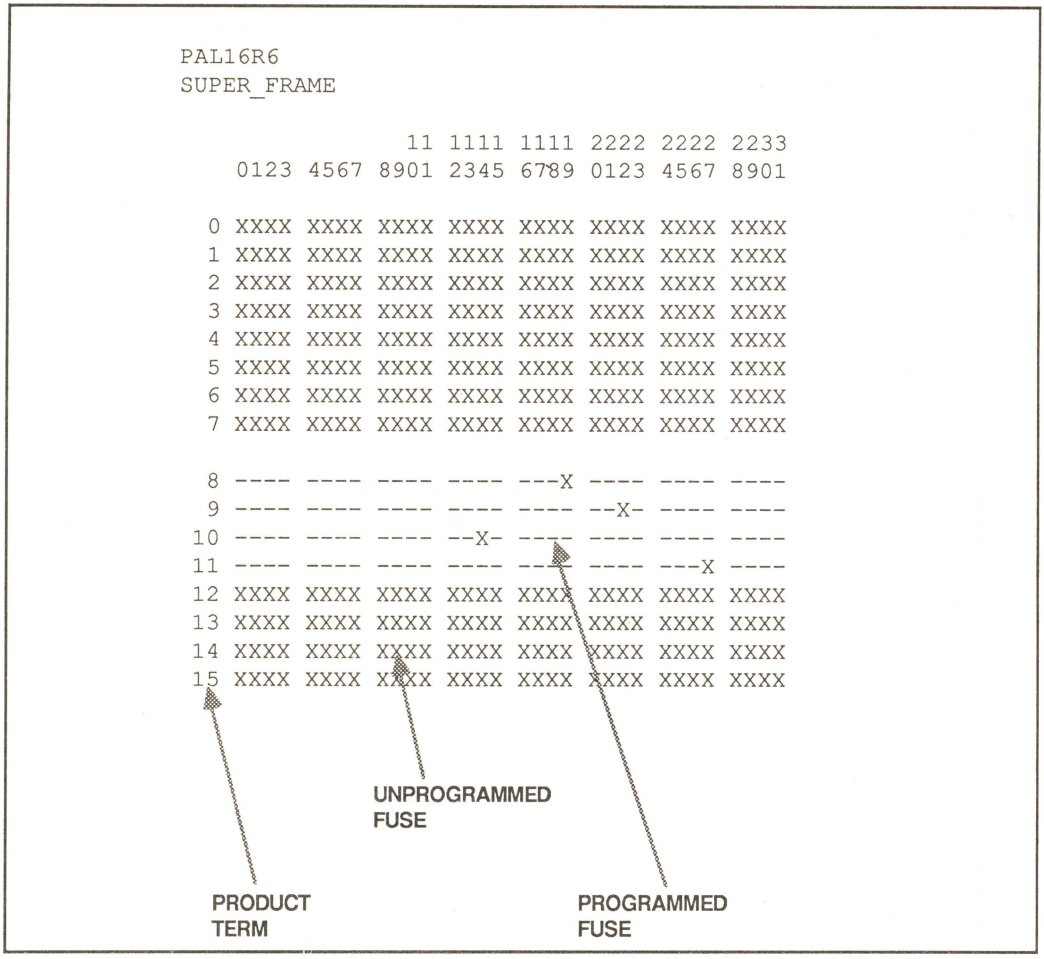



Figure 4-19

The SUPER.XPT Fuse Map

3.10.2

## View The JEDEC File

The JEDEC file is programmer-readable and should be downloaded to the device programmer. Refer to *Program The Device*, Chapter 7, for more information. Figure 4-20 shows a sample JEDEC file.

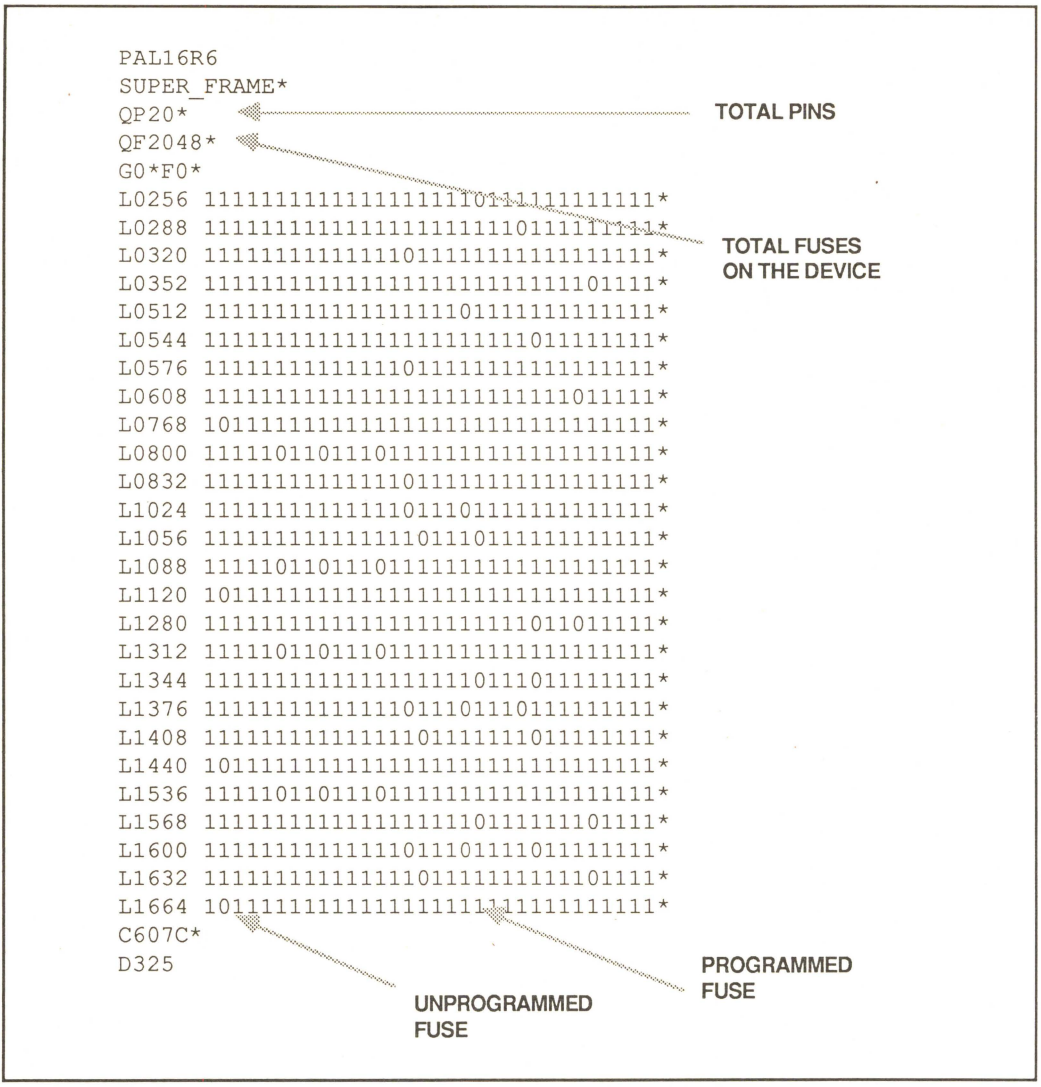



Figure 4-20

The JEDEC Fuse Data from SUPER.JED

3.11

## Run The Software From DOS

PALASM 2 software can be run directly from DOS. Call the programs from DOS by entering the program names instead of using the PALASM menu. The following procedure briefly describes how the programs are invoked directly.

### Create The Input File

Create the input file using any text editor. Remember to include the extension .PDS in the input filename. Refer to *Build The Boolean Equation Design*, Chapter 4, or *Build The State Machine Design*, Chapter 5 for detail on how to create the design that becomes your input file.

### Check The Syntax

The following procedure describes how to run the syntax check program.

1. Insert the disk containing the executable files in drive B. Insert the disk containing your PDS file in drive A.

   Make sure that the operating system is looking at both drives for command files. The MS-DOS command

   ```
   PATH A:\;B:\;
   ```

   takes care of this requirement. (If you are using a hard disk, specify drive C instead of B.)

2. Enter `PARSE FILENAME.PDS`<return>.

The software checks the syntax of the design file and creates an intermediate file, PALASM2.TRE, on the default drive. It also creates a PARSE.LIS file that contains the input file error messages.

### Expand The Input Equations

Enter `EXPAND` <return>.

## Minimize The Input Equations

Enter `MINIMIZE` <return>.

## Assemble the Input File

Enter `XPLOT` <return>.

## Build Simulation

Enter `SIM` <return>.

## Additional Processing Options

Enter the following program names to perform additional processing.

| | |
|---|---|
| `JEDMAN` | JEDEC disassembly |
| `TREPL2` | TRE file disassembly |

You are now ready to build your own design.

| *To...* | *Proceed to ...* |
|---|---|
| Build a Boolean equation design | Chapter 4 |
| Build a state machine design | Chapter 5 |

# 4. Build a Boolean Equation Design

## *About This Chapter*

This chapter guides you through building a Boolean design for PALASM 2 software.

| *To . . .* | *Refer to Section . . .* |
|---|---|
| Build a Boolean equation design | 4.1 |
| Determine the polarity of an output | 4.2 |
| Tailor the design for specific devices | 4.3 |
| Verify your design with a checklist of guidelines | 4.4 |

4.1

## Build A Boolean Equation Design

A Boolean equation design specifies logic functions for programming a device to perform specified tasks and give specified outputs. The design is constructed using a text editor and must contain only ASCII characters. Store the file using the filename format of FILENAME.PDS. PALASM 2 software interprets the design and translates it into a JEDEC file for downloading to a device programmer.

Figure 4-21 shows the structure of the design.

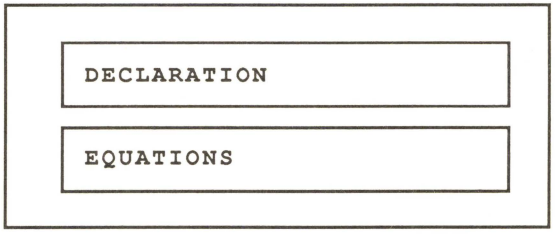



**Figure 4-21**

**Structure of the Boolean Equation Design**

Table 4-6 describes each segment of the design.

**Table 4-6**

**Description of Boolean Equation Design Segments**

| Segment | Description |
|---|---|
| DECLARATION | Design identification, device and pin data, string substitutions |
| EQUATIONS | Boolean functions that define outputs in terms of inputs and feedback, and equations that define programmable functions |

"Equations" is a reserved word. The software reserves certain words to identify design segments and information, device codes, commands, functions, and pin defaults. Do not use the reserved words for any other purpose. Some reserved words are keywords that identify the block of information that follows. All reserved words are listed in *General Syntax,* Section 4.1.1, item 5.

The general syntax rules discussed in the following section must be observed to build a Boolean design.

4.1.1

## General Syntax

The following general syntax rules apply to building the Boolean input file.

1. Maximum line length is 128 characters or columns. Data beyond the limit must be placed on the next line.

2. Characters are upper or lower case alphanumeric, spaces, tabs, and underscores. Tabs are translated as spaces. Unless otherwise stated, *never* use ` ~ ! @ # $ % ^ & - {} [] " ? < or >

3. Table 4-7 lists characters that perform special functions. Do not use these characters for any other purpose.

**Table 4-7**

**Special Characters and Functions**

| Character(s) | Function |
|---|---|
| ' ' | (Single quote or apostrophe) Delimits string characters to be substituted |
| , | Pin list separator |
| ( ) | Enclose pins in logic expressions |
| ; | Precedes comments (text the software does not see). **Extensive commenting is a good habit**. Comments may start anywhere on the line and must be preceded by a semi-colon ( ; ). |

Table 4-7 (Continued)

Special Characters and Functions

| Character(s) | Function |
|---|---|
| / | NOT or active-low value |
| * | AND |
| + | OR |
| :+: | XOR |
| = | Combinatorial output equation operator |
| *= | Latched output equation operator |
| := | Registered output equation operator |
| Operator precedence: / * + :+: | |

4. For a programmable polarity part, a pin with the same polarity in the pin list and in the equation has an active-high output. A pin with different values in the pin list and the equations has an active-low output. For more information, refer to *Polarity,* Section 4.2.

5. PALASM 2 software reserves the following words to identify design segments and information, device codes, commands, functions, and pin defaults:

| | |
|---|---|
| AUTHOR | DEFAULT_BRANCH |
| BEGIN | DEFAULT_OUTPUT |
| CHECK | DO |
| CHIP | ELSE |
| CLKF | END |
| CLOCKF | EQUATIONS |
| CMBF | FOR |
| COMPANY | GND |
| CONDITIONS | HOLD_STATE |
| DATE | IF |

5. Reserved words (continued):

| | |
|---|---|
| MASTER_RESET | RSTF |
| MEALY_MACHINE | S |
| MOORE_MACHINE | SETF |
| NC | SIMULATION |
| NEXT_STATE | STATE |
| OR | STRING |
| OUTPUT_ENABLE | THEN |
| OUTPUT_HOLD | TITLE |
| PATTERN | TRACE_OFF |
| POWER_UP | TRACE_ON |
| PRLDF | TRST |
| R | VCC |
| REVISION | WHILE |

4.1.2

## Build The Declaration Segment

Information in the Declaration segment helps document the design before processing. It also defines pin names and string substitutions. This segment appears first in the design as shown previously in Figure 4-21. Figure 4-22 shows the keywords and information structure of the segment.

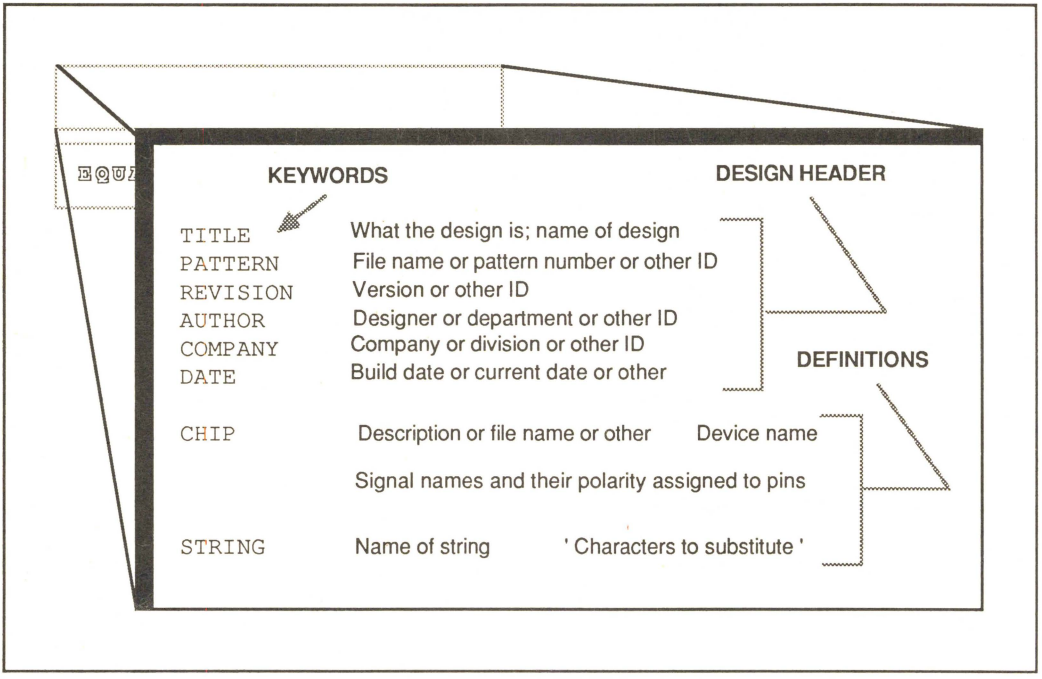



Figure 4-22

Structure of the Declaration Segment

The design header helps document and identify the design. The software inserts the header into the output files to help identify them also. You may omit any part of the Declaration segment except the CHIP keyword and definition; PALASM 2 software requires them to process the design. When any part of the design header is omitted, the software issues a warning message during assembly and continues processing.

Information for the design header contains up to 24 significant characters after the keyword and extra blank spaces. Characters beyond 24 are truncated during processing. Definitions for CHIP and STRING have the special syntax described below.

4.1.2.1

### CHIP Syntax

The CHIP definition is required to process the design. It provides information about the device and pins. CHIP syntax requires three entries:

*Syntax*

CHIP Description or file name or other Device name

Signal names and values assigned to pins (pin list)

Figure 4-23 lists each entry, describes its specific syntax and shows an example.

**1 Description or filename or other**

1 letter followed by up to 13 alphanumeric characters

**2 Device name**

Any device supported by the software (Chapter 1 contains the complete list of devices)

```
CHIP       NOT_REAL_1              PAL16R8

;PINS  1       2       3      4      5     6     7     8     9     10
       CLOCK   DCLOCK SEN1   SEN2   I2    /I3   /14   I5    /I6   GND

;PINS  11    12    13       14    15    16    17    18    19    20
       SDI   NC     RESET   SDO   TOP1   BOT1 TOP2 BOT2 MID   VCC
```

**3 Signal names and polarity assigned to pins (pin list)**

Sequential list of signal names and their polarities for the pins of the device. Begin a name with a letter and follow it with up to 13 alphanumeric characters. Place a slash ( / ) before the name to indicate active-low; otherwise, the signal is active-high. Separate names by commas, blank spaces, or carriage returns. Label unused pins NC (no connect). GND and VCC are reserved words and must be placed on ground and power pins.

**Figure 4-23**

**CHIP Syntax and Pin List**

Notice the commented lines of pin numbers in Figure 4-23. Commenting the pin numbers helps identify signal names and pin numbers for writing equations. Using mnemonic phrases or names for pins (e.g., CLOCK for the clock pin) also makes writing equations easier and helps document the design.

4.1.2.2

## STRING Substitution Syntax

Substituting a frequently used string of characters with an identifier is optional. You may know from the design's purpose, the device logic diagram, and the CHIP definition which combinations of pins will be used frequently before writing any equations. You can add string definitions to the Declaration segment as the design progresses. If you use strings, two entries must follow the STRING keyword:

***Syntax***

STRING String name ' Characters or previously defined string names to substitute '

The single quotes ( ' . . . ' ) are delimiters that identify the characters for substitution. Figure 4-24 lists each entry and its specific syntax and shows two examples.

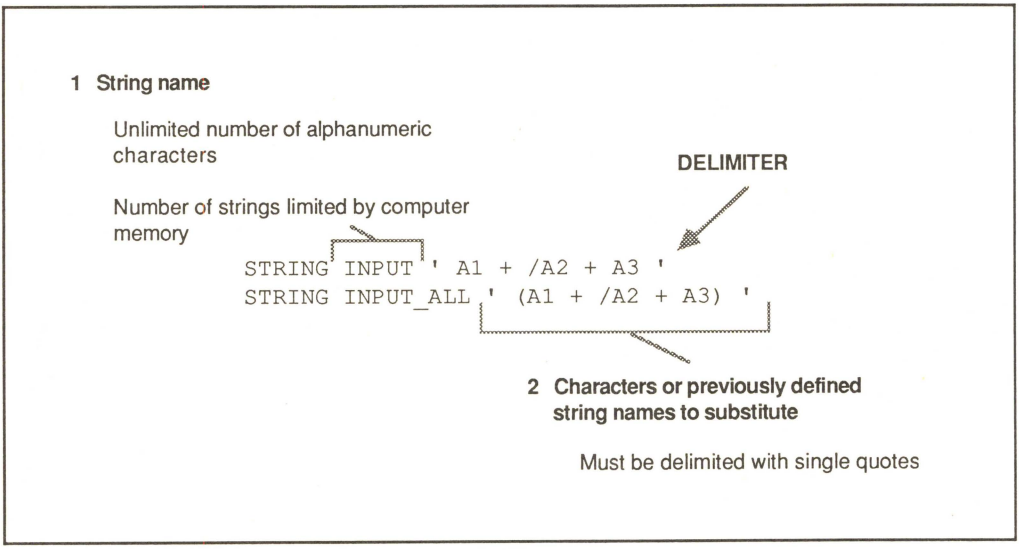



Figure 4-24

STRING Information and Syntax

In Figure 4-24, notice that parentheses were added to the INPUT string to form the string entitled INPUT_ALL. The difference between how /INPUT and /INPUT_ALL are compiled is shown in Table 4-8.

**Table 4-8**

**Compilation of String Definitions in Figure 4-24**

| String Term | Software Compilation |
|---|---|
| /INPUT | /A1 + /A2 + A3 |
| /INPUT_ALL | /(A1 + /A2 + A3)<br>= /A1 * A2 * /A3 |

Figure 4-25 shows a sample Declaration segment.

```
EQUA

TITLE     NOT_REAL_6
AUTHOR    J. ENGINEER
CHIP      NOT_REAL_6              PAL16R4

;PINS 1       2       3     4     5  6     7  8   9   10
      CLOCK   DCLOCK  SEN1  SEN2  A1 /A2   A3 GND NC  GND

;PINS 9    10 11   12   13    14    15   16 17 18  19  20
      SDI  NC RST  SDO  TOP1  BOT1  MID  NC NC NC  NC  VCC

STRING INPUT ' A1 + /A2 + A3 '
STRING INPUT_ALL ' (A1 + /A2 + A3) '
```

**Figure 4-25**

**Sample Declaration Segment**

4.1.3

## Build The Equations Segment

The Equations segment contains Boolean functions and equations for programmable functions that define outputs in terms of inputs and feedback. The equations determine which fuses are programmed.

The keyword

```
EQUATIONS
```

is required to identify this segment of the design.

The syntax of an equation depends on the function it performs. The following sections discuss the purpose and syntax of combinatorial, registered, latched, and functional equations.

4.1.3.1

## Combinatorial Equations

Combinatorial equations combine signals for immediate output:

***Syntax***

```
Output_Pin  =  Signal * Signal * . . .
            +  Signal * Signal * . . .
            +  . . .
```

The combination of signals on the right of the = define the output signal on the left. This output signal can be active-high (Output_Pin) or active-low (/Output_Pin). (*Polarity,* Section 4.2, contains more details about polarity.). Figure 4-26 shows a sample CHIP entry (not part of the Equations segment) and combinatorial equations.

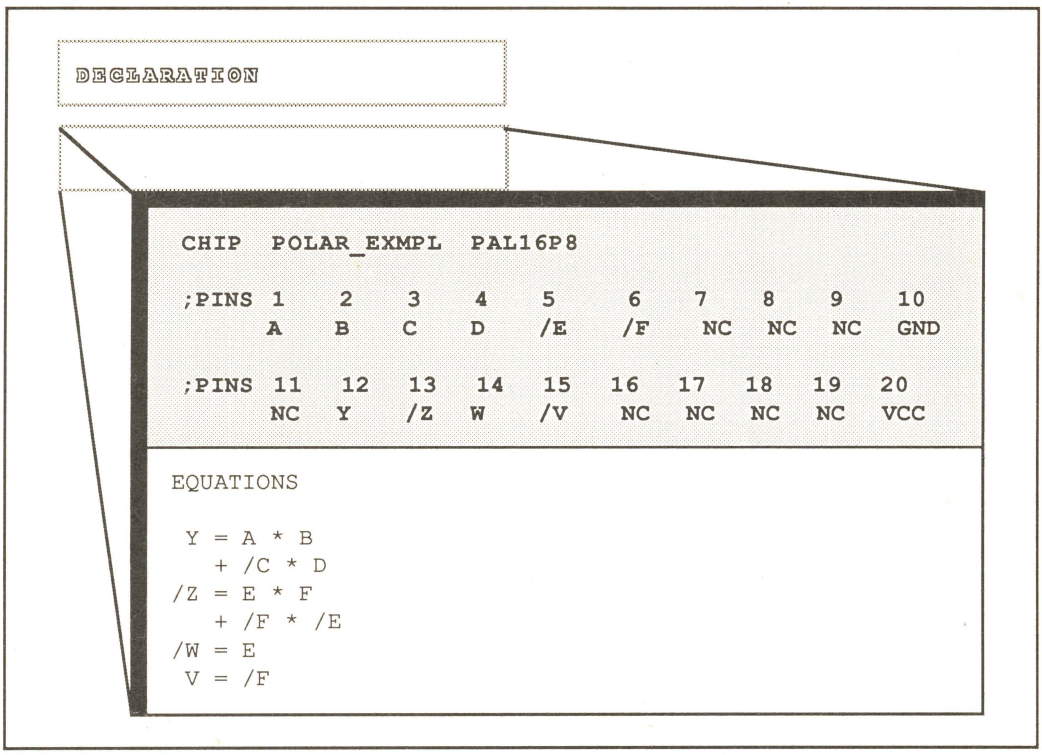



Figure 4-26

Sample CHIP Entry and Combinatorial Equation

On devices with programmable polarity, the polarity fuse is programmed or left intact according to the polarities given on the left side of the equation and those defined in CHIP. When these two polarities are the same, the fuse is programmed, giving an active-high output. When the two polarities differ, the fuse is left intact, leaving the output active-low. *Polarity,* Section 4.2, contains more discussion.

In Figure 4-26, equations for outputs Y and Z have the same polarity as in the pin list, indicating an active-high output. On the programmable-polarity PAL16P8, the outputs will be programmed as active-high. On the active-low PAL16L8, these equations would cause an error because an active-high output is not allowed. Specifying active-low outputs for the active-high PAL10H8 would also cause an error.

4.1.3.2

## Registered Equations

Registered equations generate logic functions for devices with registered outputs. For example, each output of the PAL16R8 device is a registered output:

***Syntax***

```
Output_Pin :=  Signal * Signal * . . .
           +   Signal * Signal * . . .
           +   . . .
```

The combination of signals on the right of := defines the next value of the output signal on the left. This output signal can be active-high (Output_Pin) or active-low (/Output_Pin). (*Polarity,* Section 4.2, contains more details about polarity.) Figure 4-27 shows a sample CHIP entry (not part of the Equations segment) and registered equations.

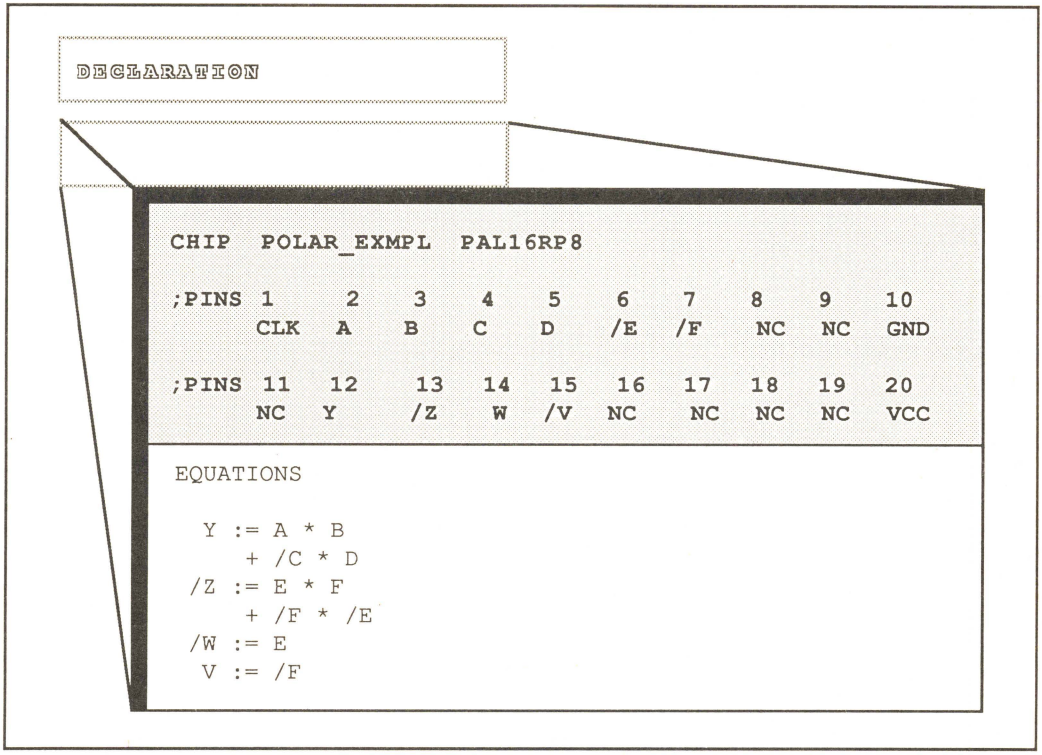



Figure 4-27

Sample CHIP Entry and Registered Equation

In most cases, the clock to the register is a dedicated clock pin. For example, on the PAL16R8, pin 1 is the clock pin. On the PAL20RA10, a special product term generates the clock. (Refer to *PAL16RA8 And PAL20RA10 Special Considerations,* Section 4.3.5, for more information.)

The transition at the output of the register takes place on the rising edge of the clock. This output signal can be active-high (output) or active low (/output).

4.1.3.3

## Latched Equations

Latched equations generate logic functions for devices with latched outputs. For example, each output of the PAL10H20G8 device may be used as a latched output:

*Syntax*

```
Output_Pin  *=  Signal * Signal * . . .
            +   Signal * Signal * . . .
            +   . . .
```

The signals on the right of *= define the output pin on the left. This output signal can be active-high (Output_Pin) or active-low (/Output_Pin). (*Polarity,* Section 4.2, contains more information.)

4.1.3.4

## Functional Equations

Functional equations define these special programmable functions:

- Clock (PAL16RA8 and PAL20RA10 only; refer to *PAL16RA8 And PAL20RA10 Special Considerations,* Section 4.3.5.)
- Set
- Reset
- Three-state
- Registered/Combinatorial output selection (PAL32VX10 only; refer to *PAL32VX10 Special Considerations,* Section 4.3.7)

Some devices offer individually programmable output functions. Individually programmable functions can be defined for each output. The PAL16RA8 and PAL20RA10, for example, are individually programmable output devices.

Refer to *Tailor the Design for Specific Devices,* Section 4.3, for more details about devices with programmable outputs. The following sections discuss globally programmable set and reset equations, and individually programmable three-state equations.

### The Programmable Set And Reset Functions

The set function creates a logic 1 at a register; the reset creates a logic 0.

When set and reset are globally programmable, a fictional 25th pin must be named after VCC in the pin list:

***Example***

```
;PINS     18     19     20     21     22     23     24     25
          OUT1   OUT2   OUT3   OUT4   OUT5   OUT6   VCC    GLOBAL
```

The options for specifying global set and reset in the Equations segment are:

***Syntax***

Options for set:

| | |
|---|---|
| 25th_Pin.SETF = GND | (Default: always disabled) |
| 25th_Pin.SETF = VCC | (Always set) |
| 25th_Pin.SETF = One_Product_Term | |

When the product term is true, the registers are set.

Options for reset:

| | |
|---|---|
| 25th_Pin.RSTF = GND | (Default: always disabled) |
| 25th_Pin.RSTF = VCC | (Always reset) |
| 25th_Pin.RSTF = One_Product_Term | |

When the product term is true, the registers are reset:

***Example***

```
GLOBAL.SETF = A * /B
GLOBAL.RSTF = /A * B
```

Individually programmable set and reset for the PAL16RA8 and PAL20RA10 are discussed in *PAL16RA8 And PAL20RA10 Special Considerations,* Section 4.3.5.

## The Programmable Three-State Function

When the device has individually programmable three-state, any outputs may be put in a logic off (high impedance) state:

***Syntax***

| | |
|---|---|
| Output_Enable_Pin.TRST = VCC | (Always enabled) |
| Output_Enable_Pin.TRST = GND | (Default unless an equation is defined for that pin) |
| Output_Enable.TRST = One_Product_Term | (User-defined) |

Enabled means visible output (three-state buffer is high); disabled means no output (three-state buffer is low). This syntax applies whether the outputs are registered or combinatorial:

***Example***

```
O14 := A * /B
O14.TRST = C * /I4
```

The output is enabled when the three-state equation product term is true.

4.2

## Polarity

Some devices have fixed active-high or active-low outputs. Some devices have programmable output polarity. To achieve the desired polarity on an output, the signals in your design must be defined correctly. The factors that determine the polarity of an output are:

- Whether the device has fixed or programmable output polarity
- Whether the polarity of the output pin name in the pin list and on the left side of an equation operator is the same or opposite

4.2.1

### Programmable Polarity

**Note:** To define polarity for the PAL32VX10, refer to *PAL32VX10 Special Considerations,* Section 4.3.7.

The relationship between the signal in the pin list and the signal in the Boolean equation has a direct bearing on the polarity of the output pin. To achieve the desired programmable output polarity, you must define the signals in the pin list and Boolean equation appropriately.

When the polarities of a signal in the pin list and in the equation are the same, the output polarity is active-high. When the polarities of a signal in the pin list and in the equation are different, the output polarity is active-low. As an example of programmable polarity, Figure 4-28 shows that for pin 12, while the signal in the pin list is active-low (/O1), the signal in the equation is active-high (O1). This programs the output polarity for pin 12 as active-low (/O1) because the two signals have different polarities.

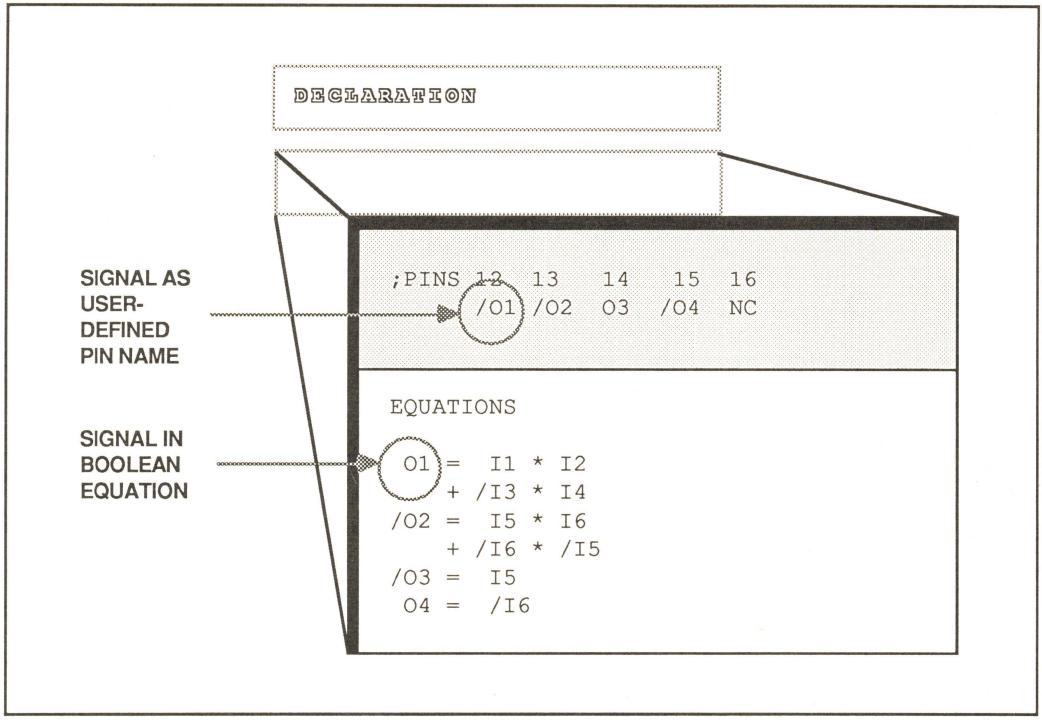



**Figure 4-28**

**Comparison of Polarity in Pin List and Equations**

Table 4-9 summarizes the polarity of signals for active-low output.

Table 4-9

Summary of Signals for Active-Low Output

| Output Polarity | Pin List | Boolean Equation |
|---|---|---|
| Active-low<br>/O1 | Active-low<br>/O1 | Active-high<br>O1 |
| | OR | |
| | Active-high<br>O1 | Active-low<br>/O1 |

Table 4-10 lists the output polarity for the four possible combinations of signals in the pin list and the equations.

Table 4-10

Table for Determining Output Polarity

| | | Signal in Boolean Equation | |
|---|---|---|---|
| | | S | /S |
| Signal in Pin List | S | HIGH | LOW |
| | /S | LOW | HIGH |
| | | Output Polarity | |

Using Table 4-10 makes defining equations simple. If we want the output polarity to be active-high, one possible combination is to use /S in the pin list and /S in the Boolean equation.

Figure 4-29 summarizes all possible pin list and equation polarity combinations for a partial pin list.

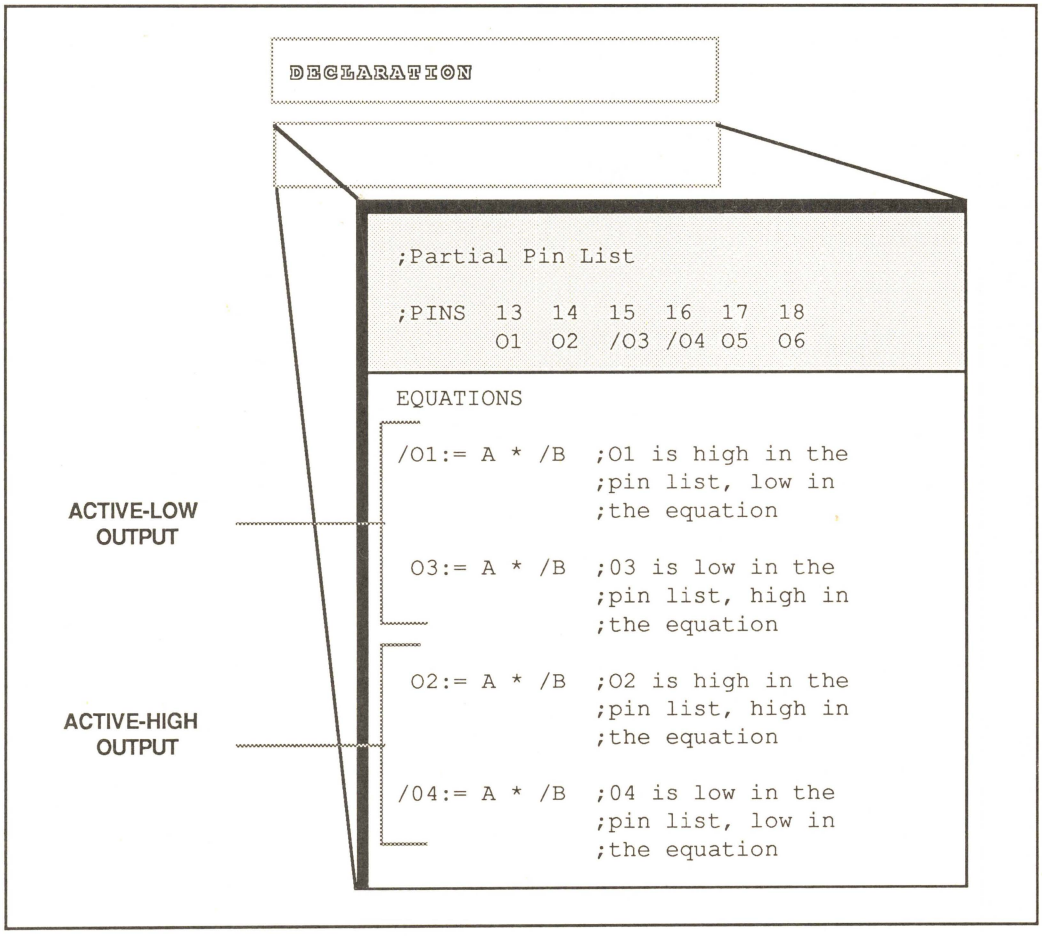



**Figure 4-29**

**Summary of Output Polarity for Programmable Polarity Parts**

**Note:** It is important to remember that on some programmable polarity parts, the polarity fuse is located before the register. It does not affect the set or reset function of the output. If no equation is defined for an output, the polarity fuse is left intact.

4.2.2

## Fixed Polarity

While any combination in Figure 4-29, works on a programmable polarity device, fixed polarity active-low devices, such as PAL16L8, do not accept the same polarity in both the pin list and the Boolean equation. The combinations these devices accept are /S in the pin list and S in the equation or vice versa for a fixed active-low output.

For a fixed active-high output device, the polarities in the pin list and in the equation must be the same (S . . . S or /S . . . /S). If active-high output is specified for an active-low output device, or if active-low output is specified for an active-high output device, errors occur.

4.3

## Tailor The Design For Specific Devices

This section provides general considerations for designing with PLS and PAL devices, and specific considerations for designing with the following devices:

- PAL10H20G8
- PAL16RA8 and PAL20RA10
- PAL22V10
- PAL32R16
- PAL32VX10

Only the special functions and features not discussed previously are included in this section.

4.3.1

## PLS Device General Considerations

When designing with any PLS device, consider the following items.

1. PLS devices have S-R flip-flops instead of D flip-flops. Each output equation requires two equations, one for S and one for R:

   ***Syntax***

   ```
   Output_Pin.S := Product_Terms
   Output_Pin.R := Product_Terms
   ```

   The total supply of product terms for the device determines the number of product terms for each equation because the device allows product term steering. The polarity of the output pin in the equations must be the same as declared in the pin list.

2. PLS devices have buried register nodes. A buried register can be used for feeding back a signal into the array; it cannot send its output to an output pin. Assign a name to each node after VCC in the pin list beginning with the earliest node:

   ***Example***

   ```
   CHIP INPUT_OUTPUT PLS105

   CLK I2 I3 I4 I5 I6 I7 I8 I9 O10 O11 NC NC GND
   O13 O14 O15 O16 OE I18 I19 I20 I21 I22 I23 NC NC VCC
   P0 P1 P2 P3 P4 P5
   ```

   Write equations for the buried nodes using the same syntax as for output pins.

3. PLS devices have a complement array for inverting signals, which can be used to save product terms. Assign a name for the complement array after the buried nodes:

   ***Example***

   ```
   CHIP INPUT_OUTPUT PLS105

   CLK I2 I3 I4 I5 I6 I7 I8 I9 O10 O11 NC NC GND
   O13 O14 O15 O16 OE I18 I19 I20 I21 I22 I23 NC NC VCC
   P0 P1 P2 P3 P4 P5 COMP
   ```

   If you want to use the complement array, define it once in the Equations segment:

***Example***

```
/COMP = I2 * I3
      + I4 */I5
```

Notice that the polarity of the complement array on the left side of the equation must be the opposite of the polarity defined in the pin list. You may use it as a term in the equations as often as needed:

***Example***

```
O11.S := COMP * I6 * I7
      + I9 */I10
```

Notice that when you use the array as a term of an expression, it has the same polarity as in the pin list.

4. PLS devices have a dedicated programmable pin for preset or output enable (PR/OE). It controls either the asynchronous preset of all registers to high or controls the three-state output buffers of the output registers. By programming the PR/OE pin to control the three-state output buffers, the preset function is permanently disabled. Otherwise, the default is that preset is programmed.

   To program the pin as preset, define a SETF functional equation for any or all buried nodes and outputs:

   ***Syntax***

   Any_Output_Pin.SETF = Name_of_PR/OE_Pin
   Any_Buried_Node.SETF = Name_of_PR/OE_Pin

   ***Example***

```
O15.SETF = OE
P1.SETF = OE
```

   The PR/OE pin must be asserted high to activate the preset. The output pin or buried node cannot have a slash, / , in front of it.

   To program the pin as output enable, define a three-state functional equation for any or all outputs:

***Syntax***

Any_Output_Pin.TRST = Name_of_PR/OE_Pin

***Example***

```
O11.TRST = /OE
```

The PR/OE pin must be asserted low to enable outputs. The output pin or buried node cannot have a slash, / , in front of it.

**Note:** Define .SETF or .TRST; not both.

4.3.2

## PAL Device General Considerations

The following general considerations apply to PAL devices.

1. The number of product terms is fixed for each output. Therefore, the fixed number of terms allotted to that output determines the number of product terms allowed for the equation for that output.

   If the device has product term steering, then the number of product terms for an equation is limited by the number of terms allotted to each pair of outputs. For example, if a pair of outputs has eight product terms, you can use five for one output and three for the other.

2. Run the EXPAND program to translate XOR gates into AND and OR gates or to expand nested parentheses.

4.3.3

## PAL10H20G8 Special Considerations

1. The PAL10H20G8 has programmable latched or combinatorial outputs.

2. To use a PAL10H20G8 I/O pin as an input, refer to *Controlling Output Enable With SETF* in Section 6.2.1.2.

   Remember that the symbol *= is the latched equation operator.

3. On the PAL10H20G8, pins 1 and 13 may be used as a latch enable or as regular inputs.

4.3.4

## PAL22V10 Special Considerations

When designing with the PAL22V10 device, consider the following items.

1. The three-state function can be controlled using an individually programmable three-state function for each output (refer to *Functional Equations,* Section 4.1.3.4 for the syntax).

   The default for a combinatorial equation with no .TRST function is VCC.

2. You can program combinatorial or registered equations by using the = or := operators to define the outputs.

3. To use an I/O pin permanently as an input, do not use the pin as an output on the left side of the equation. Use the pin only as an input on the right side of the equation operator to define other outputs.

4. Pin 1 may be used as a clock and as an input for combinatorial or registered output in the same design.

5. The synchronous global preset sets all registers to high. Define a fictional 25th pin after VCC in the pin list to serve as the global preset. Define this function once in the design:

   ***Syntax***

   25th_Pin.SETF = One_Product_Term

   Refer to *Controlling Output Enable With SETF* in Section 6.2.1.2 for how to control this function in simulation.

6. The asynchronous global reset sets all registers to low. Define a fictional 25th pin after VCC in the pin list to serve as the global reset. Define this function once in the design:

   ***Syntax***

   25th_Pin.RSTF = One_Product_Term

   Refer to *Controlling Output Enable With SETF* in Section 6.2.1.2 for how to control this function in simulation.

4.3.5

## PAL16RA8 And PAL20RA10 Special Considerations

In addition to the programmable set, reset, and three-state functions, the PAL16RA8 and PAL20RA10 offer a programmable clock. The programmable clock function allows you to clock individual output signals. The software indicates an error if a clock function is defined for a combinatorial output because combinatorial output has no clock. If no clock function is defined for a registered output, the software issues an error.

Table 4-11 shows the syntax for PAL16RA8 and PAL20RA10 functional equations.

Table 4-11

Syntax for PAL16RA8 and PAL20RA10 Functional Equations

| Function | Syntax | Example |
|---|---|---|
| Set | Output_Pin.SETF = VCC (Bypass register with reset also high) | |
| | Output_Pin.SETF = GND (Default) | |
| | Output_Pin.SETF = One_Product_Term | `OUT.SETF = A * /B` |
| Reset | Output_Pin.RSTF = VCC (Bypass register with set also high) | |
| | Output_Pin.RSTF = GND (Default) | |
| | Output_Pin.RSTF = One_Product_Term | `OUT.RSTF = /A * B` |
| Clock | Output_Pin.CLKF = GND (Default: no clock) | |
| | Output_Pin.CLKF = Name_of_Pin_Used_as_Clock | `OUT.CLKF = CLK` |
| | Output_Pin.CLKF = Input_Product_Term | `OUT.CLKF = A * B * /C` |
| Three-state | Output_Pin.TRST = VCC (Default: output enabled) | |
| | Output_Pin.TRST = GND (Default unless an equation is defined) | |
| | Output_Pin.TRST = One_Product_Term | `OUT.TRST = C * /I4` |

You can bypass the register by asserting both set and reset high in two different ways. One way is to be explicit for each registered output:

*Example*

```
OUT := A + /B + D * E        ;Output defined as registered
OUT.SETF = VCC               ;SETF always true
OUT.RSTF = VCC               ;Reset always true
OUT.CLKF = GND
```

The other way is to be implicit by writing a combinatorial equation for the output:

```
OUT = A + /B + D * E         ;Output defined as
                             ;combinatorial - no clock
```

In the implicit case, the software automatically assigns the set and reset functions to VCC and the clock function to GND.

In some cases, you might want to use the register without the set and reset functions:

***Example***

Being explicit:

```
OUT := A + /B
OUT.SETF = GND
OUT.RSTF = GND
OUT.CLKF = CLK
```

Being implicit:

```
OUT := A + /B
OUT.CLKF = CLK
```

The software assigns the set and reset functions to GND by default.

**Note:** If you define output as combinatorial ( = ), the default value for both set and reset is VCC. If you define output as registered ( := ), the default value for both set and reset is GND.

4.3.6

## PAL32R16 And PAL64R32 Special Considerations

You can program the outputs of the PAL32R16 as registered or combinatorial, in banks of eight. The equation operator determines the function. If you use := , the output is registered; if you use = , the output is combinatorial. All outputs within a bank must be configured the same way.



4.3.7

## PAL32VX10 Special Considerations

In addition to global set and reset functions, the PAL32VX10 has associated with it the following architectural features unique in the PAL device family:

- Each register can be buried so that its contents cannot be observed directly on the output pin.
- Each output has an internal exclusive-OR gate which can either be used as such or as a polarity inverter, depending on the application. The XOR gate also allows the user to create D, T, J-K, or S-R flip-flop types. In addition, you can set outputs as registered or combinatorial (and set the register type) dynamically, specifying the option you want with a product term.

To use these special features, observe the following special rules.

1. The pin list for a PAL32VX10 includes the expected pin names, with VCC followed by a fictional pin name used to specify global set and reset functions. This is followed by ten names that specify the nodes located at the /Q outputs of the internal registers at pins 14 through 23:

   ***Example***

   ```
   CHIP INPUT_OUTPUT PAL32VX10

   CLK I2 I3 I4 I5 I6 I7 I8 I9 I10 I11 GND
   I13 O14 O15 O16 O17 O18 O19 O20 O21 O22 O23 VCC
   GLOBAL R14 R15 R16 R17 R18 R19 R20 R21 R22 R23
   ```

   The relationship between the internal registered node names and output names in the pin list above is that R14 corresponds to O14.

2. When you want the output to be a registered function of product terms, you must define it this way at the buried registered node. You must then define the output pin as the internal register. If you want the register to be used only for feedback, do not define the output pin as the internal register:

   ***Syntax***

   ```
   Internal_Register := Sum_of_Product_Terms
   Output_Pin := Internal_Register
   ```

   or

   ```
   /Internal_Register := Sum_of_Product_Terms
   Output_Pin := /Internal_Register
   ```

   depending on your polarity preference. **Notice that the internal register always has the same polarity in both equations.**

   ***Example***

   ```
   R14 := I2 * I3
   OUT14 := R14
   ```

   **Note:** If you want the output from the buried register to be visible, the output pin must be defined as a function of the buried register.

3. When you want the output to be a combinatorial function of product terms, define the output node with the = operator. In this case, the buried registered node must not be defined. Use the regular combinatorial equation ( = ) notation.

4. Because there is only one AND/OR array for each output, you can define either the internal node or the output node as a sum of products, but not both. Follow rules 2 and 3 above for defining registered or combinatorial outputs to meet this requirement.

5. The PAL32VX10 has only one exclusive-OR gate per output. Each output equation can therefore contain at most one exclusive-OR function (of two terms). If you use the exclusive-OR gate in your equation, then you must not also use it as a polarity inverter. Thus, if you define an internal node or output node with an exclusive-OR, its polarity must be opposite of that given in the pin list:

   ***Example***

   ```
   /R14 := I2 :+: I6 * I5 + I7 * /I8
   ```

   if R14 is the name used in the pin list.

6. **Defining output polarity for the PAL32VX10 is currently different from other devices. Observing the following guidelines will provide upward compatibility with future PALASM 2 software.**

   The polarities of the output/output feedback path and the register feedback path are determined as a pair. The polarity of one path depends on the polarity of the other path. Figure 4-29A shows the PAL32VX10 macrocell with the signal and path names that this discussion uses.

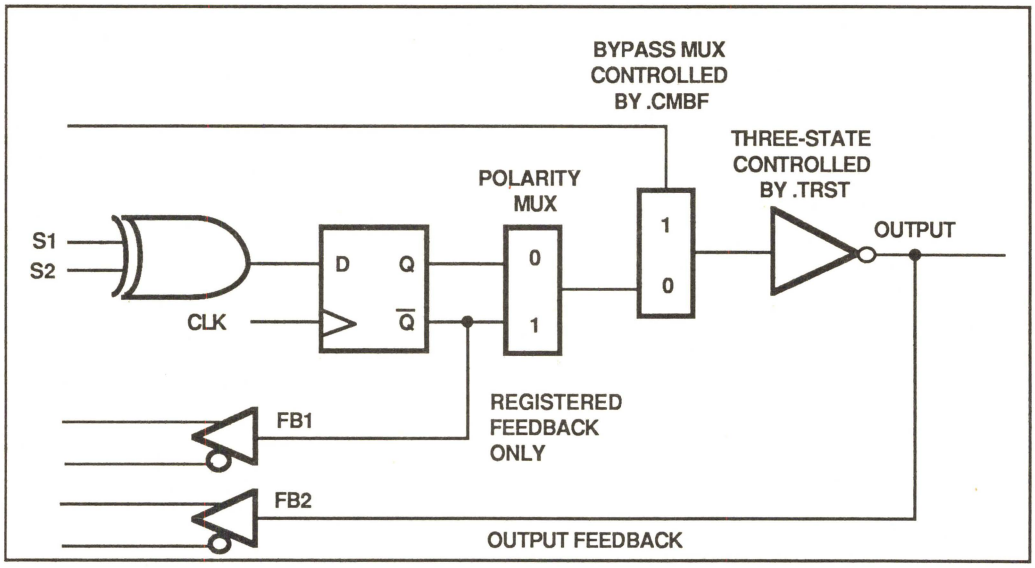



Figure 4-29A

The PAL32VX10 Macrocell

Table 4-12 lists the four polarity pairs of the output/output feedback and the register feedback paths for signal S2.

Table 4-12

Polarity Pairs of Output/Output Feedback and Register Feedback Paths for Signal S2

| Output/Output Feedback (FB2) | Register Feedback (FB1) |
|---|---|
| S2 | S2 |
| S2 | /S2 |
| /S2 | /S2 |
| /S2 | S2 |

You have only one option for selecting each pair of paths (for example, S2,S2). The option comprises two factors:

- how the output pin is defined in the pin list and in the equaton
- how the register is defined in the pin list and the equation

Table 4-13 lists the four options for path polarities for signal S2 and the corresponding pin list/equation polarity option for output signal O and register signal R. Please use **only** these combinations to program the paths.

**Table 4-13**

**Options for Path Polarities and Specifying Output Pin Polarity for Signal O and Register Polarity for Signal R**

| Option | FB2 | FB1 | Output/ FB2 Polarity | | FB1 Polarity | |
|---|---|---|---|---|---|---|
| | | | Pin List | Equation | Pin List | Equation |
| 1 | S2 | S2 | O | O | /R | /R |
| 2 | S2 | /S2 | /O | /O | R | /R |
| 3 | /S2 | /S2 | O | /O | /R | R |
| 4 | /S2 | S2 | /O | O | R | R |

For example, if you want the output/output feedback and register feedback polarity paths to be high, the output pin in the pin list and the equation must be high and the register in the pin list and the equation must be low.

No other options are currently available. For example, defining both the pin and the register in the pin list and the equations as high is invalid.

7. After defining an output node either as a function of a buried registered node or as a function of product terms, you can use the special function .CMBF to override the definition. CMBF allows dynamic selection of registered or combinatorial output:

***Syntax***

Output_Pin.CMBF = VCC

specifies a fixed combinatorial output.

Output_Pin.CMBF = GND

specifies a fixed registered output.

Output_Pin.CMBF = One_Product_Term

specifies a dynamically selected registered output if the result is low; or specifies a dynamically selected combinatorial output if the result is high:

***Example***

```
O14 := R14
R14 := I1 * I2 * /I3
O14.CMBF = I4
```

This is a registered output, but will be combinatorial if I4 is asserted high. You need not use the .CMBF if you do not want dynamic selection.

**Note:** You can use only one product term for this function.

**Note:** The Examples disk contains a complete PAL32VX10 design specification example.

4.4

## Checklist For Verifying Boolean Equation Designs

The following checklist helps you verify that your design meets PALASM 2 software's syntax requirements for Boolean designs.

1. Is the input file free of control characters such as form feeds, and was it created as a clean ASCII file?
2. Does the keyword CHIP appear before the design name, device type, and list of pin names?
3. Does the keyword EQUATIONS preface all Boolean equations used?
4. Have you defined all strings to be used as logic replacements for terms in the Boolean equations?

5. On 20-pin devices, is GND specified as pin 10 and VCC as pin 20? On 24-pin devices, is GND specified as pin 12 and VCC as pin 24? On 28-pin devices, is GND specified as Pin 14 and VCC as pin 28?

6. If you are specifying an active-low output on a programmable polarity device, is the signal name on the left side of the equation the logical opposite of the signal name specified in the pin list? Are the signal names the same for active-high parts?

7. Are you within the maximum number of product terms for any output?

8. Are you specifying .TRST equations for individually programmable three-state outputs with three-state buffers only?

9. Are you specifying .CLKF equations for PAL16RA8 and PAL20RA10 designs only?

10. Are all comments preceded by a semicolon ( ; )?

11. Does the last line in your input file terminate with a hard carriage return? (Omitting this carriage return will cause the program to crash.)

After verifying your design, proceed to *Build Simulation,* Chapter 6.


IMPORTANT
NOTES

# 5. Build a State Machine Design

### *About this Chapter*

This chapter guides you through creating a state diagram and building a state machine design file for PALASM 2 software. It describes state diagrams for Mealy and Moore machines, the structure and syntax of the state machine design file, and design considerations for PAL, PLS, and PROSE devices. It also reviews a simple design using a state diagram and design file.

| *To . . .* | *Refer to Section . . .* |
|---|---|
| Create a state diagram | 5.1 |
| Build a state machine design | 5.2 |
| Tailor the design for a PLS, PROSE, or PAL device state machine | 5.3 |

5.1

## Create A State Diagram

A state diagram illustrates the behavior of a state machine. The design is easy to build from a state diagram that includes:

- All states named (with or without assigned values) and connected to their next states
- The input values that cause state transitions when a clock pulse occurs
- The output values expected because of state transitions

The following sections discuss the definitions and diagrams for the two types of machines, Mealy and Moore.

5.1.1

## Create A Mealy State Diagram

A Mealy machine determines its outputs from the inputs and the present state. Figure 4-30 illustrates how combinatorial and registered Mealy output is achieved.

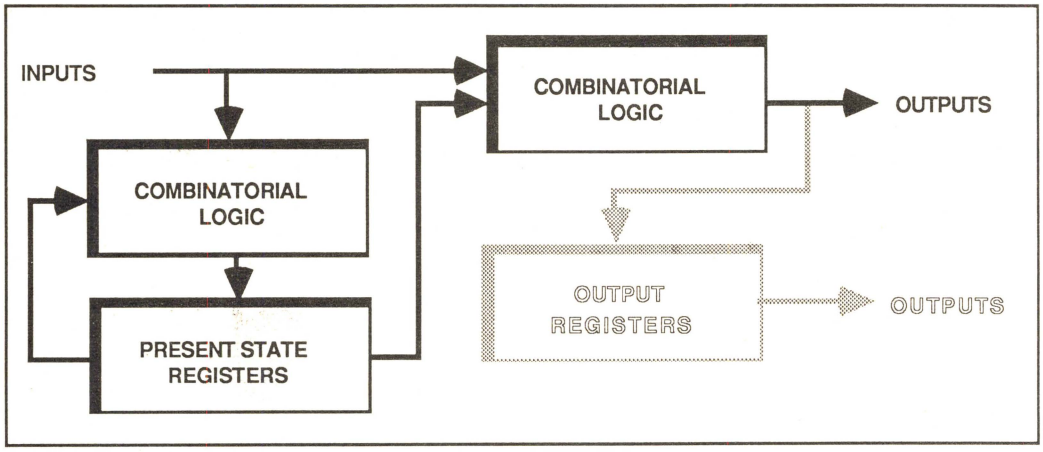



Figure 4-30

Mealy Output

In Figure 4-30, inputs from pins are combined with present state registers to determine the next state (which becomes the present state on the clock). Inputs are also combined with present state registers to determine the outputs to the pins. When outputs are combinatorial functions of inputs and the present state, the outputs are valid when the new state is reached. Registers may be added to synchronize the outputs. When outputs are registered functions, the outputs are valid one clock cycle after the new state is reached.

The state diagram for a Mealy machine reflects the independence of the states, the inputs, and the outputs. Figure 4-31 shows a functional diagram for a Mealy machine with four states.

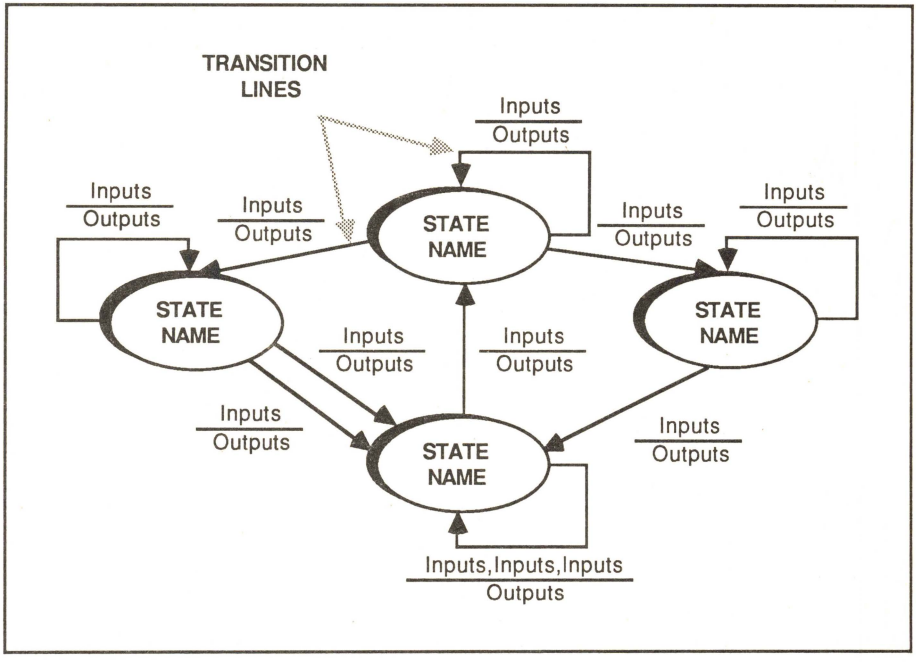



**Figure 4-31**

**Mealy Functional State Diagram**

In Figure 4-31, one or more sets of inputs and outputs are located near each transition line. You specify the input values that initiate a transition and output values that result. A set of inputs is also called a condition. Later, you will assign a name for the condition. Each transition has a unique condition; the outputs may be unique or the same. Notice

that a transition may have more than one input/output set and notice that a transition may have more than one condition (inputs) with only one set of outputs. More than one condition means that the outputs are a function of any of the true conditions.

Figure 4-32 shows sample conditions and outputs for a Mealy machine with two input and output values.

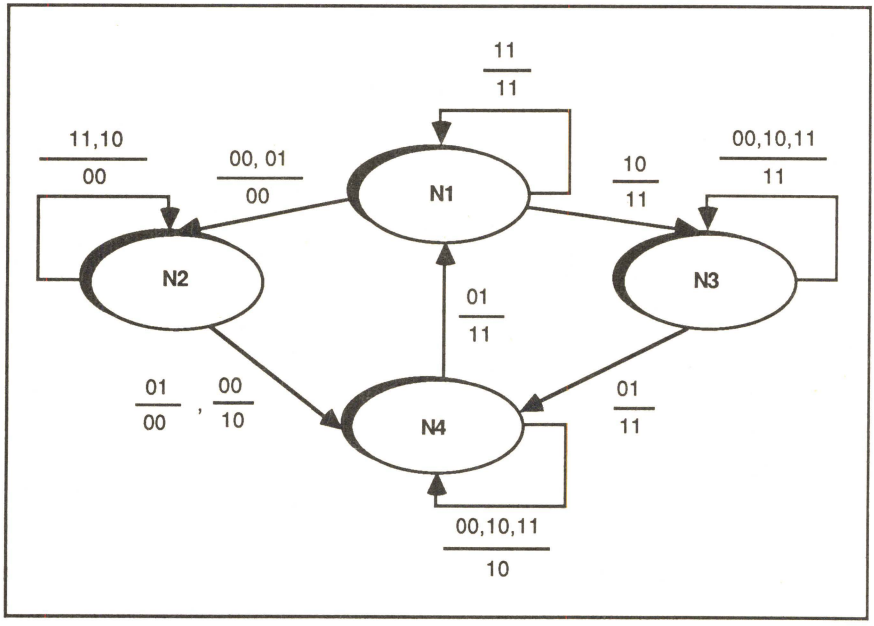



**Figure 4-32**

**Inputs and Outputs for Figure 4-31**

In Figure 4-32, when the machine is in state N1, three next states are possible:

- If the conditions are 00 or 01, the next state is N2; the outputs are 00
- If the condition is 10, the next state is N3; the outputs are 11
- If the condition is 11, the machine remains in N1; the outputs are 11

Transitions occur with the clock, after which outputs are valid. Notice in Figure 4-32 that one transition line between N2 and N4 suffices for the two lines shown in Figure 4-31. Two sets of conditions and outputs are shown because the outputs are unique.

Figure 4-32 may be simplified. One way to simplify it is to omit conditions that do not affect the transition to a new state. Outputs that do not change can also be omitted. For example, state N3 has a hold transition (N3 to N3). The hold conditions 00, 10, and 11 may be omitted from the line because they do not affect a transition to a new state. Outputs 11 may be omitted because they have not changed since the transition from N1 to N3. The hold transition at state N2 may be omitted for the same reasons.

Figure 4-33 shows the results of simplifying Figure 4-32.

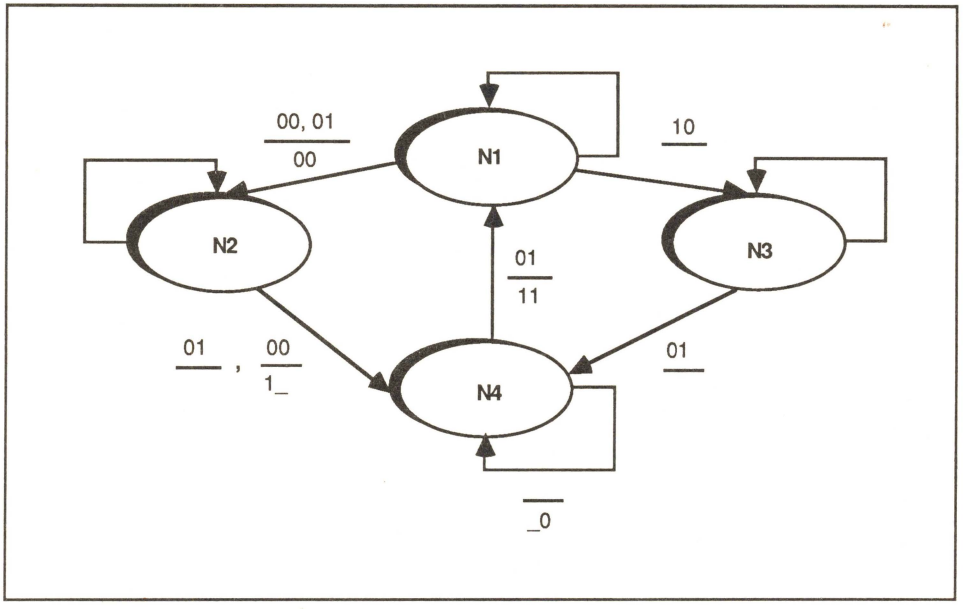



**Figure 4-33**

**Minimum Inputs and Outputs to Build a Mealy Design**

You can use other ways of simplifying state diagrams, depending on the design. For example, if outputs are high, unless otherwise specified, only conditions and low outputs need to be shown. Or, if the transition from any state is either to a next state or to an initial state, only the next states with their conditions and outputs need to be shown. The initial state can be defined separately.

The design for a Mealy machine may now be built from the information in Figure 4-33. Proceed to *Build A State Machine Design,* Section 5.2.

5.1.2

## Create A Moore State Diagram

A Moore machine determines its outputs from the present state only. Figure 4-34 illustrates how Moore output is achieved.

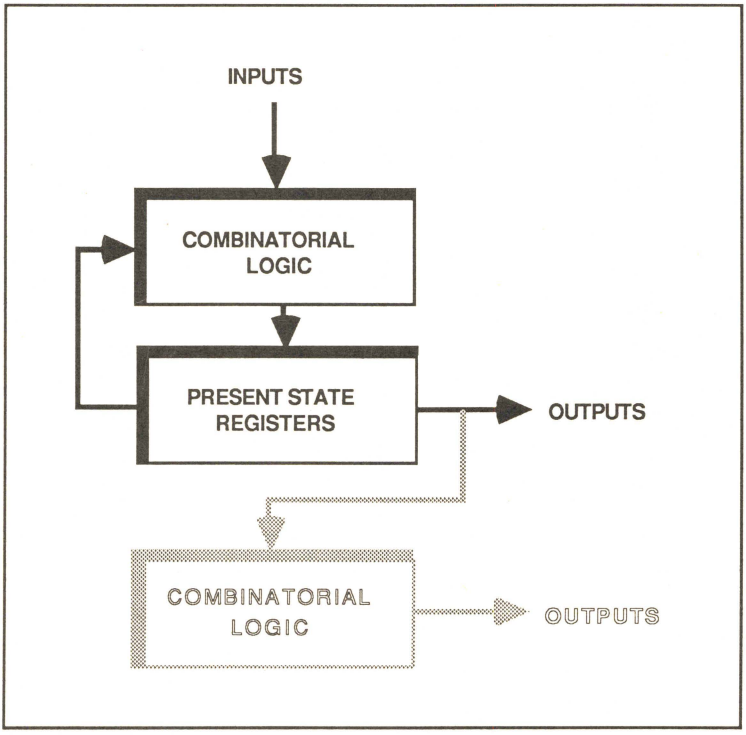



Figure 4-34

Moore Output

In Figure 4-34, inputs from pins are combined with the present state registers to determine the next state (which becomes the present state on the clock). Only the present state determines the outputs. For a Moore machine, combinatorial and registered outputs are valid when the new state is reached.

The state diagram for a Moore machine reflects the dependence of the outputs on the states. Figure 4-35 shows a functional diagram for a Moore machine with four states.

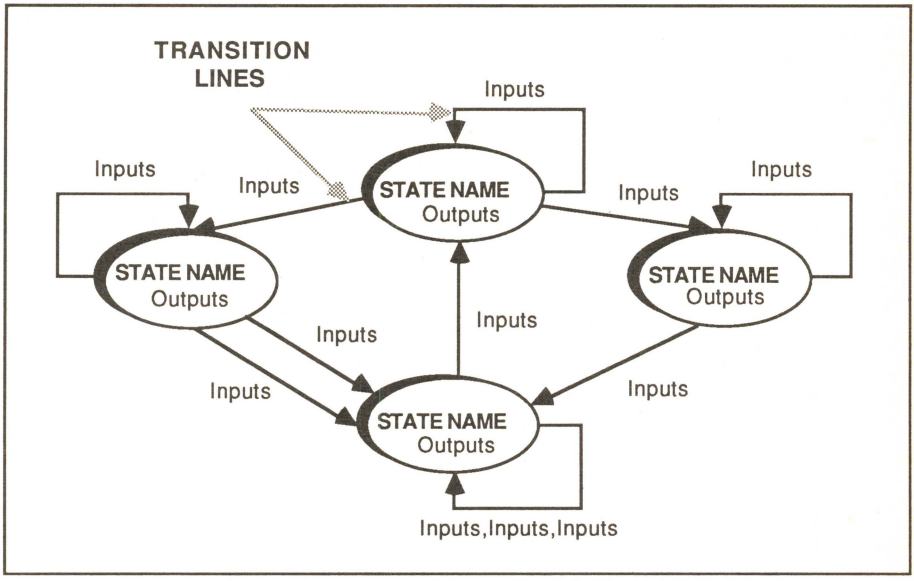



**Figure 4-35**

**Moore Functional State Diagram**

In Figure 4-35, one or more inputs are located near each transition line. You specify the input values that initiate a transition. A set of inputs is also called a condition. Later, you will assign a name for the condition. You place the outputs from each state with the state name. Notice that a transition may be caused by more than one condition. Each transition has a unique condition.

Figure 4-36 shows sample conditions and outputs for a Moore machine with two inputs and outputs.

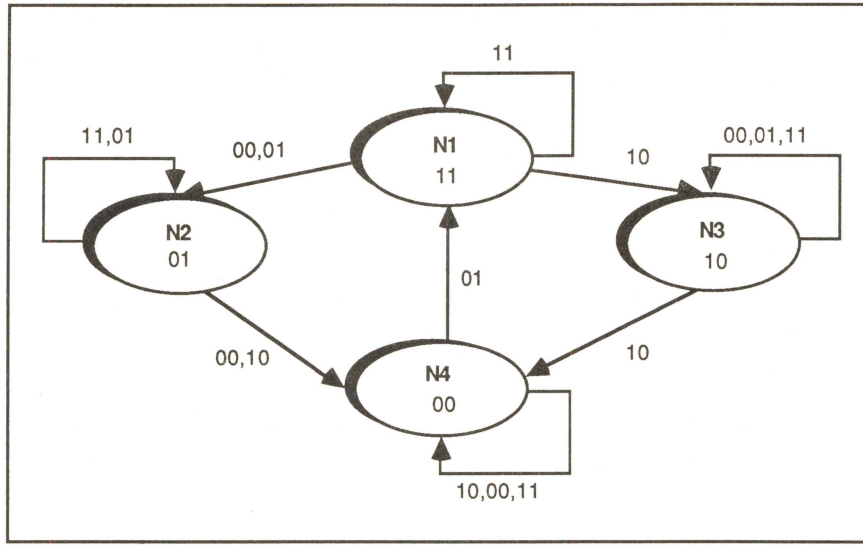



Figure 4-36

Inputs and Outputs in Moore Diagram

In Figure 4-36, when the machine is in state N1, three next states are possible:

- If the conditions are 00 or 01, the next state is N2 and the outputs are 01
- If the condition is 10, the next state is N3 and the outputs are 10
- If the condition is 11, the machine remains in N1 and the outputs are 11

Notice that one transition line between N2 and N4 suffices for the two lines shown in Figure 4-35.

Figure 4-36 may be simplified. One way to simplify it is to omit conditions that do not affect the transition to a new state. For example, N4 has a hold transition (N4 to N4). The hold conditions 10, 00, and 11 may be omitted from the line because they do not affect a transition to a new state. The hold conditions at states N1, N2, and N3 may be omitted for the same reason.

Figure 4-37 shows the results of simplifying Figure 4-36.

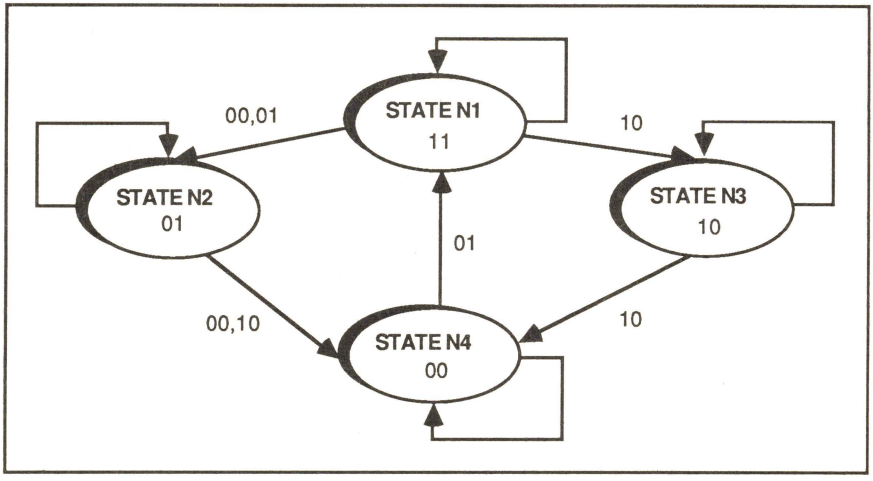



Figure 4-37

Minimum Inputs and Outputs to Build a Moore Design

As with Mealy diagrams, alternative ways of simplifying state diagrams are used depending on the design. If outputs are high, unless otherwise specified, only low outputs need to be shown. If all states have hold transitions, only the new state transitions with their conditions need to be shown.

The design for a Moore machine may now be built from the information in Figure 4-37.



5.2

## Build A State Machine Design

The PALASM 2 software design is easy to build from a state diagram. The design contains information for programming a device to cycle through defined states and give specified outputs. The design is constructed using a text editor and must contain only ASCII characters. PALASM 2 software interprets the data and translates it into a JEDEC file for downloading to a device programmer.

Figure 4-38 shows the structure of the design.

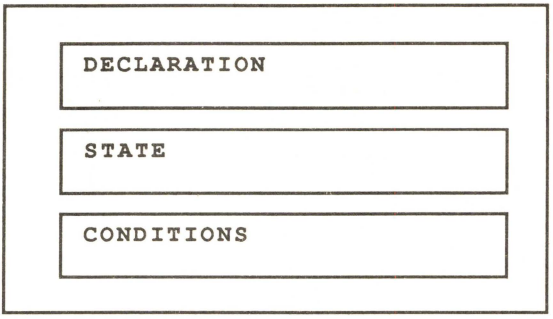



Figure 4-38

Structure of the Design

Table 4-14 describes the three segments of the design.

Table 4-14

Description of State Machine Design Segments

| Segment | Description |
|---|---|
| DECLARATION | Design identification, device and pin data, string substitutions |
| STATE | Defaults; pin assignments to states; equations for state transitions and outputs |
| CONDITIONS | Input values that determine the state transitions |

"State" and "Conditions" are reserved words. The software reserves certain words to identify design segments and information, device codes, commands, functions, and pin defaults. Do not use the reserved words for any other purpose. Some reserved words are keywords that identify the block of information that follows. All reserved words are listed in *General Syntax,* Section 5.2.1, item 5.

The general syntax rules discussed in the following section must be observed to build the design.

5.2.1

## General Syntax

The following general syntax rules apply to building the state machine design.

1. Maximum line length is 128 characters or columns. Data beyond the limit must be placed on the next line.

2. Characters are upper or lower case alphanumerics, spaces, tabs, and underscores. Tabs are translated as spaces. Unless otherwise stated, *never* use ` ~ ! @ # $ ^ & [ ] { } " ? or <

3. Table 4-15 lists characters that perform special functions. Do not use these characters for any other purpose.

**Table 4-15**

**Special Characters and Functions**

| Character(s) | Function |
|---|---|
| ' ' | (Single quote or apostrophe) Delimits string characters to be substituted |
| , | Pin list separator |
| ( ) | Enclose pins in expressions |
| ; | Precedes comments (text and characters the software does not see). **Extensive commenting is a good habit**. Comments can start anywhere on the line and must be preceded by a semi-colon ( ; ). |
| / | NOT or active-low polarity |
| % | Don't care value for global defaults |

Table 4-15 (Continued)

Special Characters and Functions

| Character(s) | Function |
|---|---|
| * | AND |
| + | OR; "Or for input condition . . ." |
| :+: | XOR (used only in condition equations) |
| -> | State transition: "Go to state . . ." |
| +-> | Local default state transition: "Otherwise, go to state. . ." |
| := | State transition and registered output equation operator |
| = | Condition, state assignment, and combinatorial output equation operator |

4. PALASM 2 software reserves the following words to identify design segments and information, device codes, commands, functions, and pin defaults:

| | |
|---|---|
| AUTHOR | DO |
| BEGIN | ELSE |
| CHECK | END |
| CHIP | EQUATIONS |
| CLKF | FOR |
| CLOCKF | GND |
| CMBF | HOLD_STATE |
| COMPANY | IF |
| CONDITIONS | MASTER_RESET |
| DATE | MEALY_MACHINE |
| DEFAULT_BRANCH | MOORE_MACHINE |
| DEFAULT_OUTPUT | NC |

4. Reserved words (continued):

| | |
|---|---|
| NEXT_STATE | SETF |
| OR | SIMULATION |
| OUTPUT_ENABLE | STATE |
| OUTPUT_HOLD | STRING |
| PATTERN | THEN |
| POWER_UP | TITLE |
| PRLDF | TRACE_OFF |
| R | TRACE_ON |
| REVISION | TRST |
| RSTF | VCC |
| S | WHILE |

5.2.2

## Build The Declaration Segment

Information in the Declaration segment helps document the design before processing. It also defines pin names and string substitutions. This segment appears first in the design as shown previously in Figure 4-38. Figure 4-39 shows the keywords and information structure of the segment.

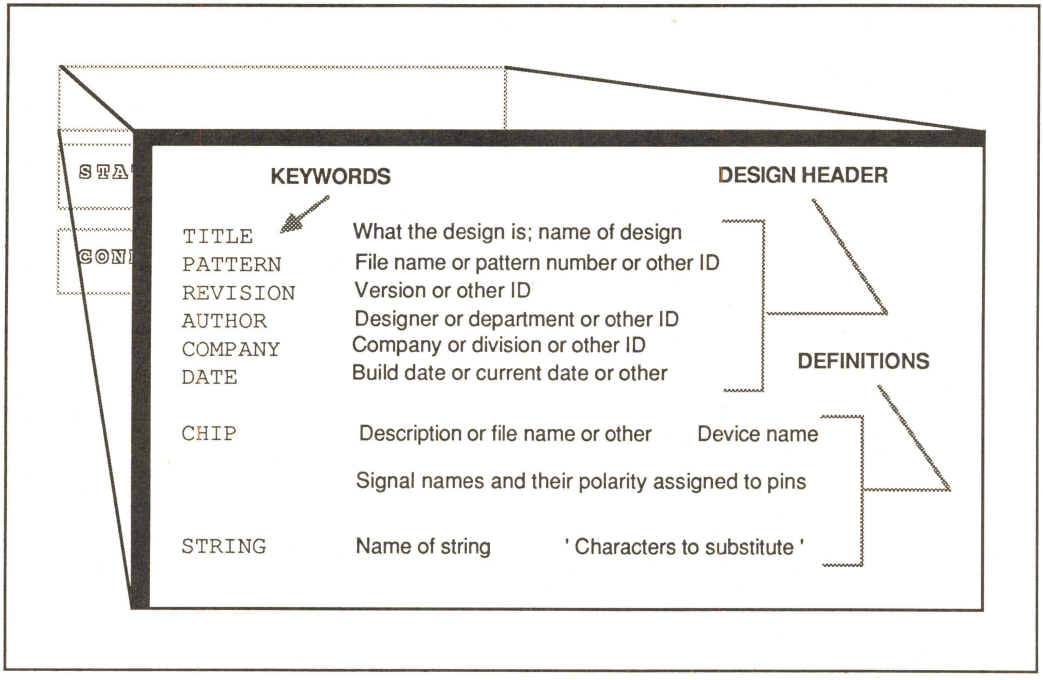



**Figure 4-39**

**Structure of the Declaration Segment**

The design header helps document and identify the design. The software copies the header into the output files to help identify them also. You may omit any part of the Declaration segment except the CHIP keyword and definition; PALASM 2 software requires the CHIP definition to process the file. When any other part of the design header is omitted, the software issues a warning message during assembly and continues processing.

Information in the design header can contain up to 24 significant characters after the keyword and extra blank spaces. Characters beyond 24 are truncated during processing.

Definitions for CHIP and STRING have the special syntax described below.

5.2.2.1

## CHIP Syntax

The CHIP definition is required to process the design. Filling in the CHIP information early makes design entry easier later. With the purpose of the design in mind and the device logic diagram in hand, fill in the three required entries for the CHIP definition:

***Syntax***

CHIP Description or file name or other Device name

Signal names and their polarity assigned to pins (pin list)

Figure 4-40 lists each entry, describes its specific syntax and shows an example.

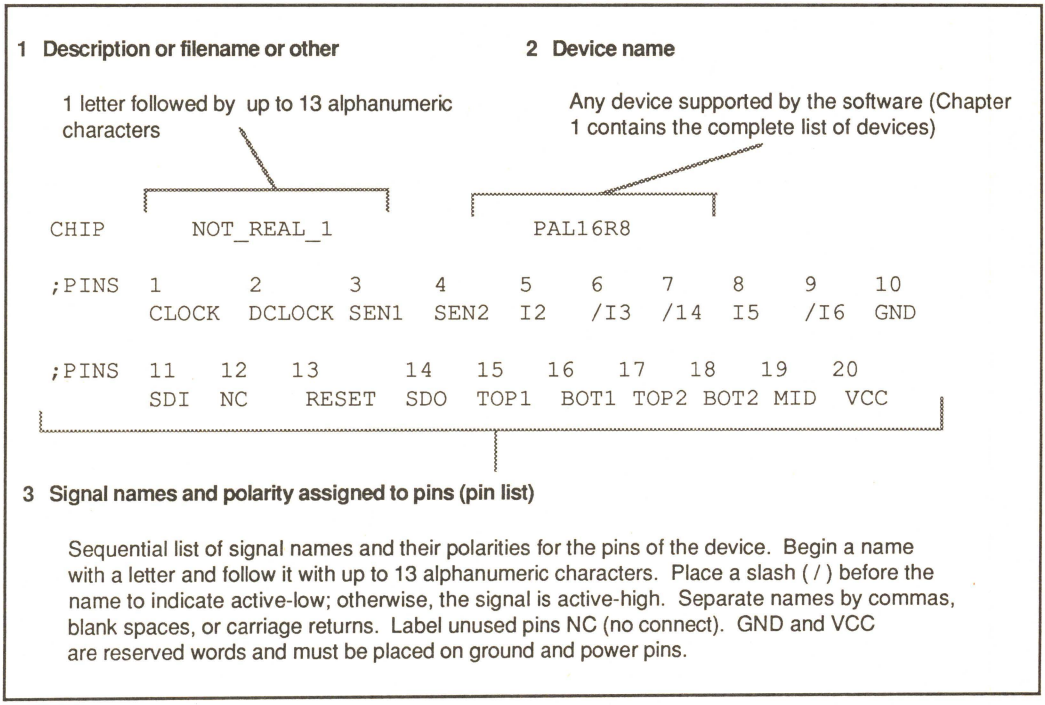



**Figure 4-40**

**CHIP Syntax and Pin List**

Notice the commented lines of pin numbers in Figure 4-40. Commenting the pin numbers helps identify signal names and pin numbers for writing state equations. Using mnemonic phrases or names for pins also makes writing equations and documenting the design easier.

5.2.2.2

## STRING Substitution Syntax

Substituting a frequently used string of characters with a short name is optional. You may know from the design's purpose, the device logic diagram, and the CHIP information which combinations of signals will be used frequently before writing equations. Otherwise, you may wish to add a string definition as the design progresses. If you use strings, two entries must follow the STRING keyword:

***Syntax***

| | | |
|---|---|---|
| STRING | String name | ' Characters or previously defined string names to substitute ' |

Figure 4-41 lists each entry and its specific syntax and shows two examples.

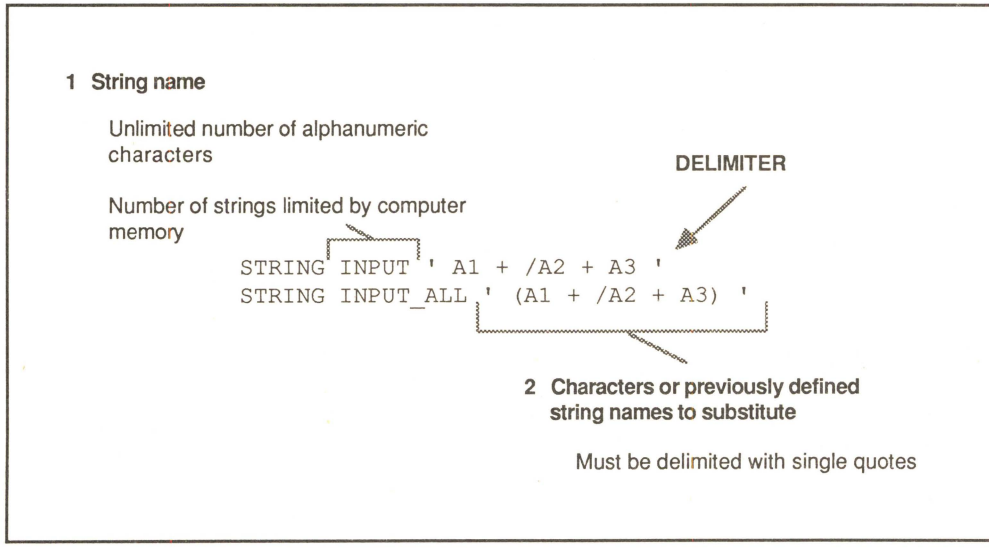



**Figure 4-41**

**STRING Information and Syntax**

The single quotes ( ' . . . ' ) in item two of Figure 4-41 are delimiters that identify the characters for substitution.

Notice that parentheses added to the INPUT string form the string entitled INPUT_ALL. The difference between how /INPUT and /INPUT_ALL are compiled is shown in Table 4-16.

**Table 4-16**

**Compilation of String Definitions in Figure 4-41**

| String Term | Software Compilation |
|---|---|
| /INPUT | /A1 + /A2 + A3 |
| /INPUT_ALL | /(A1 + /A2 + A3)<br>= /A1 * A2 * /A3 |

Figure 4-42 shows a sample Declaration segment.

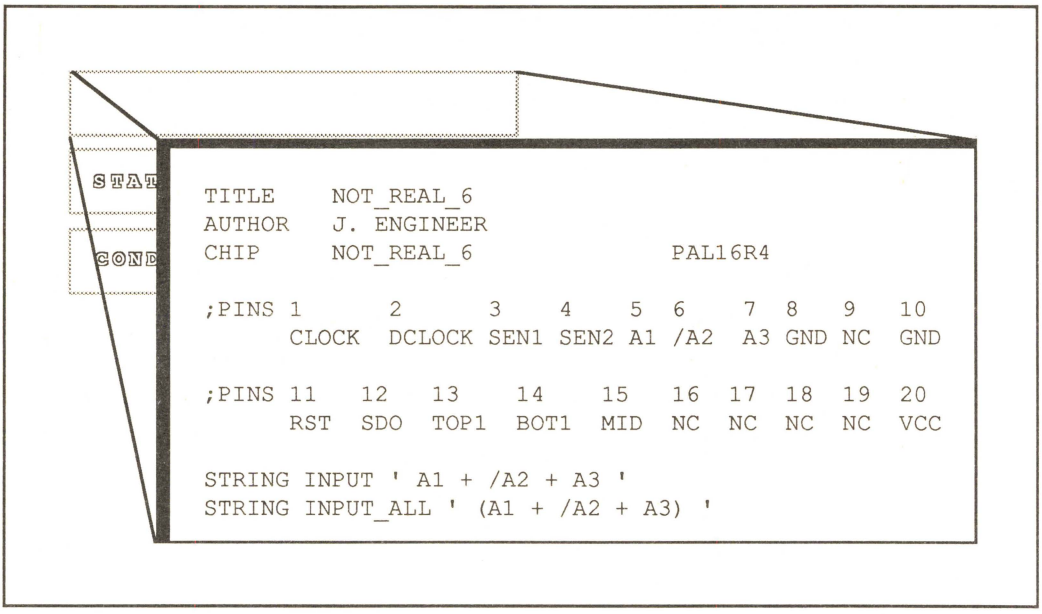



**Figure 4-42**

**Sample Declaration Segment**

5.2.3

## Build The State Segment

The State segment contains information about the design and equations that describe how the machine functions. The information comes from the state diagram and CHIP definition. The keyword

```
STATE
```

is required to identify this segment of the design. Figure 4-43 shows the information contained in this segment.

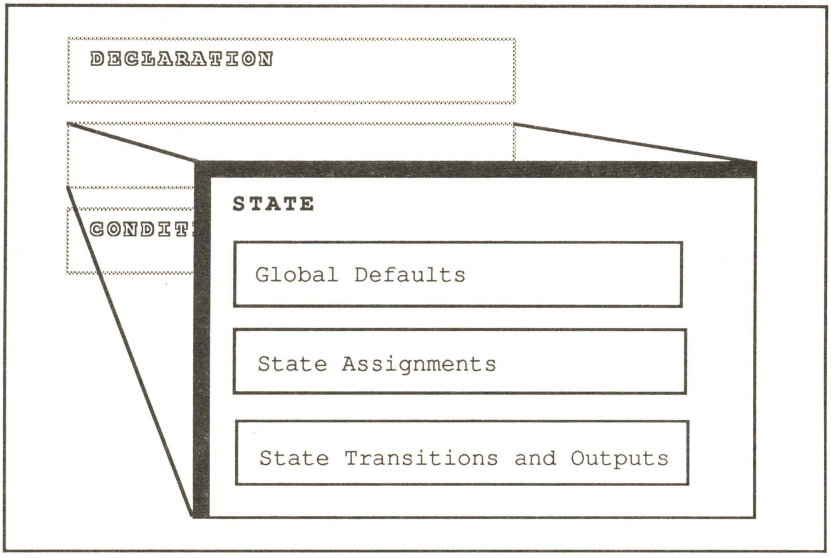



Figure 4-43

Structure of the State Segment

Table 4-17 describes the information in each box in Figure 4-43.



Table 4-17

Descriptions of State Information

| Information | Description |
|---|---|
| Global Defaults | Statements that specify the kind of machine, the outputs, and transitions when unspecified in the equations |
| State Assignments | Equations that assign pins as a bit code for each state |
| State Transitions and Outputs | Equations that specify the transitions between states and the polarity of the output signals |

The following sections discuss the purpose and specific syntax for each kind of information.

5.2.3.1

## Global Defaults

You can use defaults to speed design entry. For example, if outputs do not change on transitions, one global default option maintains the present output and you can write shorter equations. If a transition cannot be determined from the equations, another global default causes a transition to a known state.

You can default outputs to maintain the present values or to have specific values on transitions. You can default state transitions to go to the state specified, stay in the present state, or go to the next state listed in the design. Table 4-18 describes the default options.

**Table 4-18**

**Descriptions of Default Options**

| Default Options and Syntax | Description |
|---|---|
| MEALY_MACHINE (Default)<br>or<br>MOORE_MACHINE | Specifies which kind of machine the design implements. Refer to *Create A State Diagram,* Section 5.1 for definitions and state diagrams. |
| OUTPUT_HOLD Output_Pins | List of output pins that maintain their present output values when next state output values cannot be determined from PLS or PROSE device designs. Use spaces, commas, or carriage returns to separate pin names.<br><br>(Output equations are required when using OUTPUT_HOLD. Refer to *State And Output Equations* , Section 5.2.3.3.) |

Table 4-18 (Continued)

Descriptions of Default Options

| Default Options and Syntax | Description |
|---|---|
| DEFAULT_OUTPUT Output_Pins | List of output pins that default to specified values when next values cannot be determined from the design. A value is specified by placing the following special symbol before the pin name:<br><br>**Value** — **Symbol**<br>Logic 1 — None<br>Logic 0 — /<br>Don't Care — %<br><br>(Output equations are required when using this default. Refer to *State And Output Equations,* Section 5.2.3.3.)<br><br>Pin names are separated with spaces, commas, or carriage returns. |
| DEFAULT_BRANCH State_Name<br>or | Defines the next state when a next state cannot be determined from the design. |
| DEFAULT_BRANCH HOLD_STATE<br>or | Holds the machine in the present state when a next state cannot be determined from the design. |
| DEFAULT_BRANCH NEXT_STATE | Moves the machine to the state of the following equation when a next state cannot be determined from the design. The last state and output equations must define all possible transitions. |

All the default options in Table 4-18 are reserved words or keywords. Mealy machine designs that do not use defaults do not need any of the options in Table 4-18. Figure 4-44 shows sample defaults.

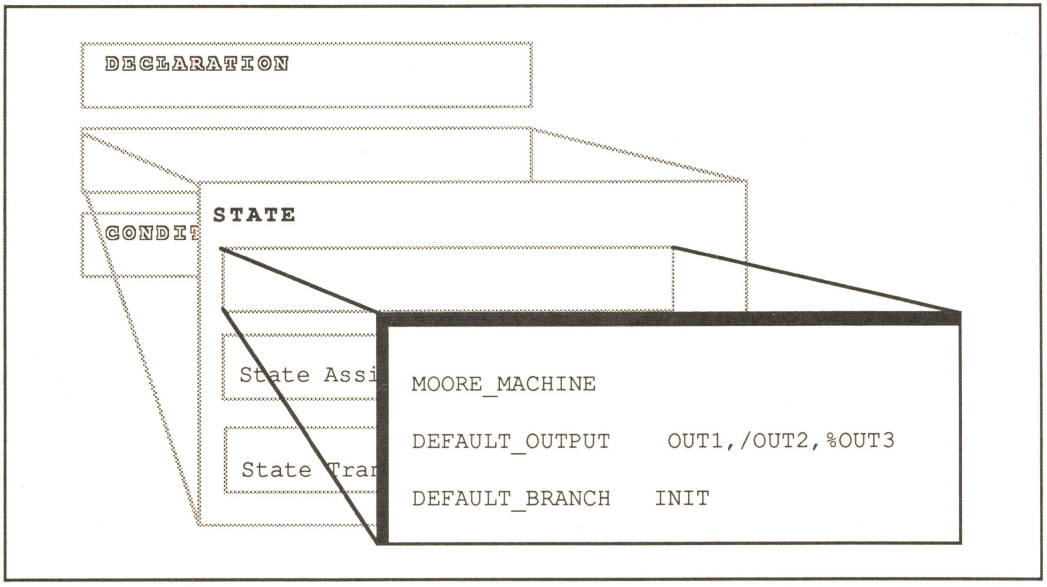



**Figure 4-44**

**Sample Defaults**

In Figure 4-44, when the outputs cannot be determined from the design, the DEFAULT_ OUTPUT definition means OUT1 will go high and OUT2 will go low; OUT3 has a don't care value. When the next state cannot be determined from the transitions, the DEFAULT_ BRANCH definition means the machine will go to state INIT.

5.2.3.2

## State Assignments

A state assignment is an equation that defines a state as a unique combination of outputs. The phrases "state bit assignment" and "bit assignment" also describe state assignments. State assignment is strongly recommended but not required. If not assigned manually, running the Expand program automatically assigns state bits for registered outputs defined as NC in CHIP. You must be careful to place NC on those outputs to be used as state bits. If you allow Expand to make assignments, however, understanding the simulation results may be difficult.

***Syntax***

State_Name = $Output_Pin_1$ . . . * . . . $Output_Pin_n$

Information for state assignments comes from the purpose of the design, the names of the states in the diagram, and the names and polarity of the output pins in the pin list. Figure 4-45 shows sample assignments.

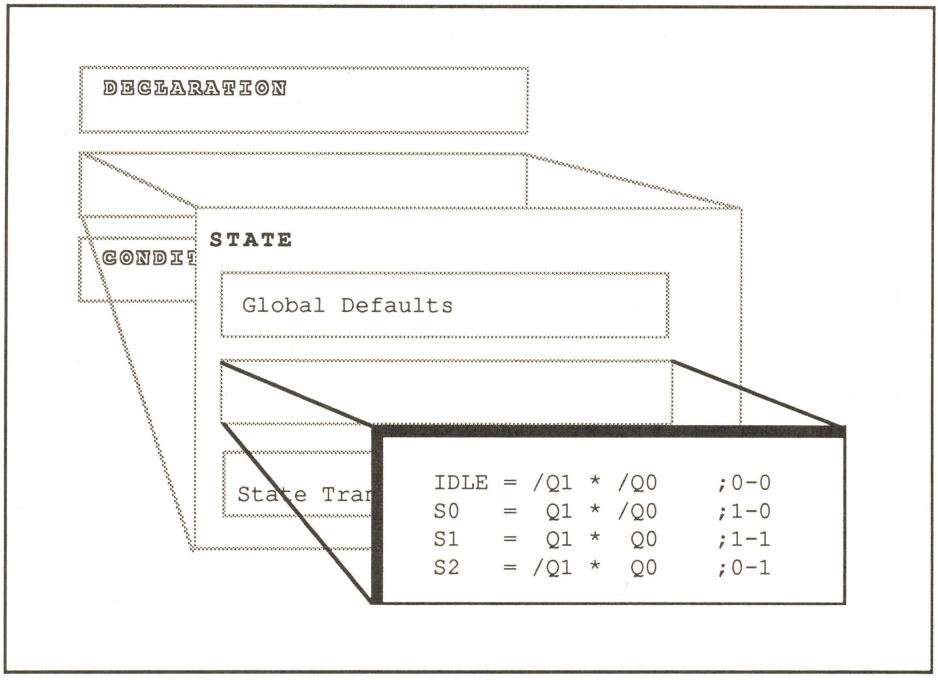



**Figure 4-45**

**Sample State Assignments**

In Figure 4-45, the machine is in state IDLE when outputs Q1 and Q0 are low. When outputs Q1 and Q0 are high, the machine is in state S1.

5.2.3.3

## State And Output Equations

State equations define the machine's sequencing in terms of conditions and next states.

Output equations define the machine's output in terms of conditions and outputs.

The state diagram and CHIP definition contain the state names and outputs for these equations.

## State Equations

A state transition equation defines states in terms of conditions that determine transitions to other states. State transitions are necessary for both Mealy and Moore designs:

***Syntax***

```
State_Name  :=   Condition1  ->  Next_State
                 . . .
            +    Conditionn  ->  Next_State
            +->  Local_Default_State
```

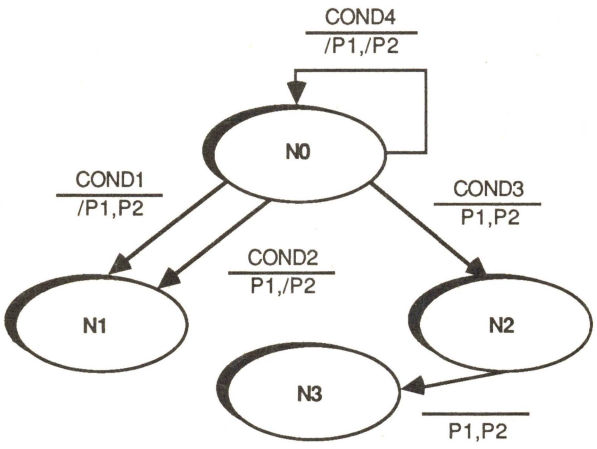
The state name corresponds to a named state in a state diagram. It must be unique and can have up to 14 alphanumeric characters. The condition is a label for the combination of input signals that determine a transition. *Build The Conditions Segment,* Section 5.2.4, discusses how to define conditions. The local default state signified by **+->** overrides the global defaults. If no local default exists, global defaults apply to unspecified transitions.

**Note:** An unconditional state transition must use VCC as a condition. The unconditional transition has the syntax

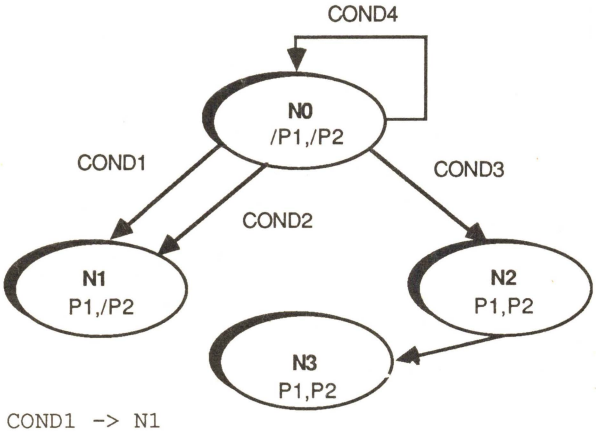
```
State_Name := VCC -> Next_State
```

Figure 4-46 contains simple state diagrams for Mealy and Moore machines and gives the state transitions that they both illustrate.

**MEALY DIAGRAM:**




**MOORE DIAGRAM:**




**STATE EQUATIONS:**

```
N0 := COND1 -> N1
      + COND2 -> N1
      + COND3 -> N2
      +-> N0
N2 := VCC -> N3
```

**Figure 4-46**

**State Equation for Mealy or Moore Machine**

Notice that in Figure 4-46 pin names (P1,P2) have replaced actual output values (0,1). Condition names (COND1,COND2,COND3) have also replaced actual input values (0,1).

The state equations for N0 and N2 are shown at the bottom of the figure. If the global default

```
DEFAULT_BRANCH HOLD_STATE
```

were specified in the Defaults segment, the local default

```
+-> N0
```

would be unnecessary.

## Output Equations

If you use OUTPUT_HOLD or DEFAULT_OUTPUT, you must have an output equation for each state equation. While a state equation specifies the conditions that cause transitions between states, an output equation specifies the conditions and the outputs from the present state.

**Note:** OUTPUT_HOLD is valid only in PLS and PROSE designs.

**Note:** If the output pins are the same as the state bits, do not use output equations and do not use OUTPUT_HOLD or DEFAULT_OUTPUT. If the output bits are different from the state bits, you must use output equations; using OUTPUT_HOLD and DEFAULT_OUTPUT is optional. Output pins not defined in the output equations or by DEFAULT_OUTPUT have the don't care value.

The syntax for output equations is different for Mealy and Moore machines. The syntax also depends on whether the outputs are registered or combinatorial:

***Syntax***

For registered Mealy output:

```
State_Name.OUTF  :=   Condition1 -> Outputs
                      . . .
                 +    Conditionn -> Outputs
                 +->  Local_Default_Outputs
```

Notice the registered equation operator :=.

***Syntax***

For combinatorial output:

```
State_Name.OUTF  =    Condition1  -> Outputs
                 . . .
                 +    Conditionn  -> Outputs
                 +->  Local_Default_Outputs
```

Notice the combinatorial equation operator =. The state name comes from the state diagram. It must be unique and may have up to 14 alphanumeric characters. The condition is a name for the combination of input signals along the transition line in the diagram. *Build The Conditions Segment,* Section 5.2.4, discusses how to define conditions. The outputs are pin names with the appropriate polarity to create the logic values found in the state diagram. The local default output signified by **+->** overrides the global defaults.

Moore output equations have the same registered and combinatorial operators but have no conditions and no local default outputs. Conditions and local defaults are not valid in Moore machine output equations because the output is determined by the state only:

***Syntax***

For registered Moore output:

```
State_Name.OUTF  :=  Outputs
```

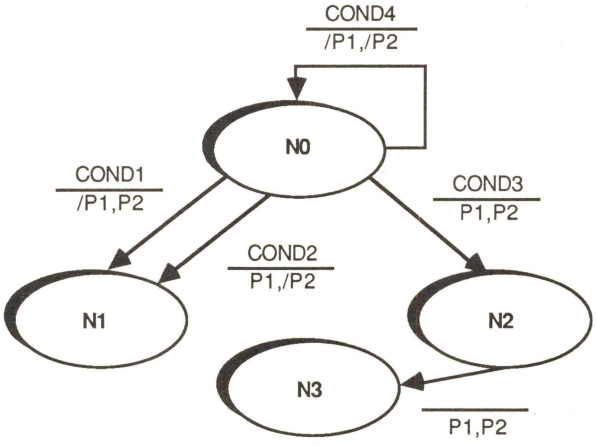
***Syntax***

For combinatorial Moore output:

```
State_Name.OUTF  =  Outputs
```

**Note:** Registered Mealy machine outputs are valid one clock cycle after the new state is reached. Mealy combinatorial outputs are valid when the new state is reached. Moore registered and combinatorial outputs are valid when the new state is reached.

Figure 4-47 contains the Mealy state diagram from Figure 4-46 with its transition and output equations.




**STATE EQUATIONS:**

```
N0 := COND1 -> N1
    + COND2 -> N1
    + COND3 -> N2
     +-> N0
N2 := VCC -> N3
```

**OUTPUT EQUATIONS:**

```
N0.OUTF := COND1 -> /P1 * P2
         -> COND2 -> P1 * /P2
         -> COND3 -> P1 * P2
          +-> /P1 * /P2
N2.OUTF := VCC -> P1 * P2
```

**Figure 4-47**

**Transition and Output Equations for Mealy Machine**

In Figure 4-47, if the global default

```
OUTPUT_HOLD /P1 /P2
```

were specified in the Defaults segment, the local default

```
+-> /P1 * /P2
```

in the output equation would be unnecessary.

Transition and output equations may be grouped together or alternated in the design. Figure 4-48 shows sample state and output equations.

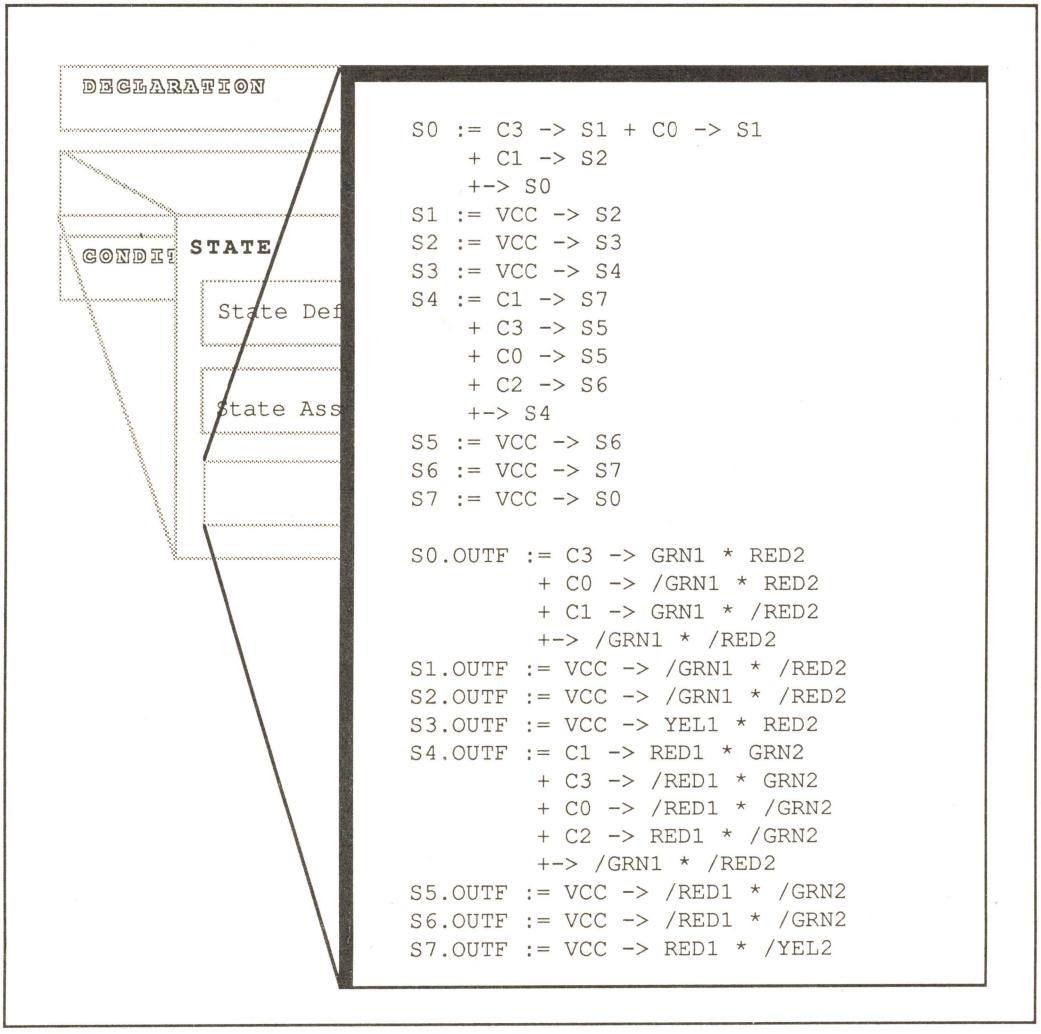



**Figure 4-48**

**Sample Transition and Output Equations**

Recall that the local default outputs for S0.OUTF and S4.OUTF,

```
+-> /GRN1 * /RED2
```

can be put in a DEFAULT_OUTPUT statement and omitted from the equations if no other global output default exists.

Figure 4-49 shows a sample State segment.

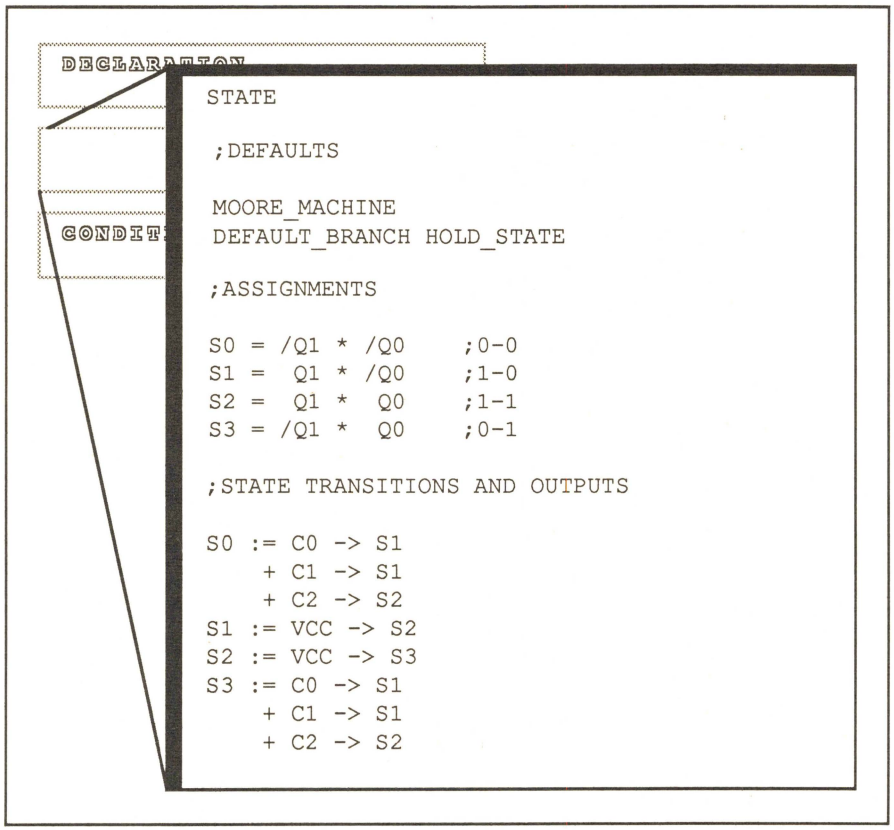



**Figure 4-49**

**Sample State Segment**

The Conditions segment follows the State segment in the design.

5.2.4

## Build The Conditions Segment

The Conditions segment contains equations that give names to unique sets of inputs. The equations identify the branching conditions used in the State segment to determine transitions and outputs.

The keyword

```
CONDITIONS
```

is required to identify this segment of the design, as shown in Figure 4-50.

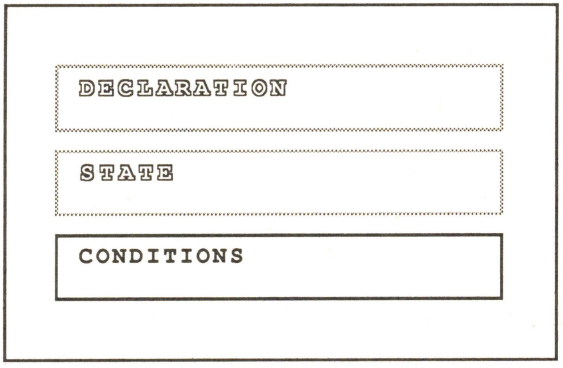



**Figure 4-50**

**Location of Conditions Segment**

The structure of a condition equation is:

***Syntax***

$$\begin{aligned} \text{Condition_Name} \quad &= \quad \text{Input}_1 \ * \ \text{Input}_x \\ &\quad \ldots \\ &+ \quad \text{Input}_y \ * \ \text{Input}_n \end{aligned}$$

The condition name must be unique and may have up to 14 alphanumerics; the number of conditions depends on the design. The inputs are the pins named in the pin list and must be unique combinatorial expressions. If a condition is only one input, a condition equation is unnecessary; use the pin name in the state and output equations. You can enclose the inputs with parentheses for DeMorgan expansion.

Illegal, conflicting conditions occur when two or more conditions may be true at the same time and are used in the same transition equation. For example, the two conditions

```
GOOD_COND1 = I1 * I2 * I3
BAD_COND2 = I1 * I4
```

used in the state transition

```
N0 := GOOD_COND1 -> N1
    + BAD_COND2 -> N2
    +-> N0
```

conflict. The state diagram in Figure 4-51 illustrates conflicting conditions.

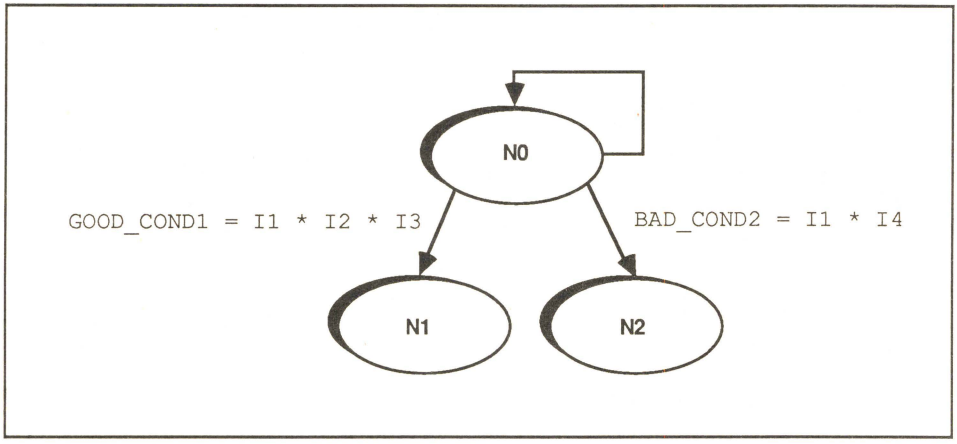



**Figure 4-51**

**State Diagram of Conflicting Conditions**

In Figure 4-51, if I1, I2, I3, and I4 are all true, the transition from N0 is random. The next state could be N1; it could be N2. PALASM 2 software issues an error message that conflicting conditions exist.

Figure 4-52 shows a sample Conditions segment with no conflicting conditions.

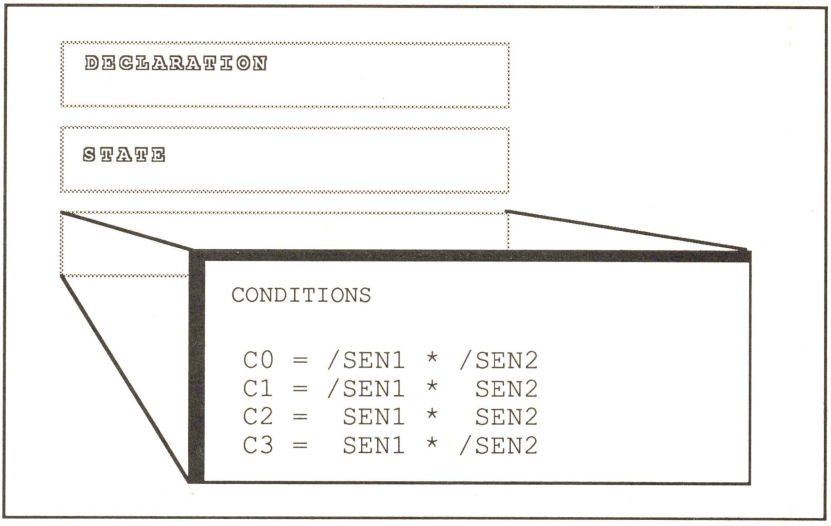



**Figure 4-52**

**Sample Conditions Segment**

Although the preceeding discussion guides you in building a PALASM 2 software state machine design, a few devices have special features that need further discussion.



5.3

## Tailor The Design For PLS, PROSE, Or PAL Device State Machines

The structure and content of the design also depends on the device you use. The following sections discuss special considerations not covered in the previous sections for building PAL, PLS, and PROSE state machine designs.

5.3.1

### PLS And PROSE Considerations

This section describes only the syntax and considerations that allow you to take advantage of special features on PLS and PROSE devices. A discussion of the syntax and structure for building a state machine design begins in *Build A State Machine Design,* Section 5.2.

Table 4-19 lists the syntax and considerations for PLS and PROSE devices.

Table 4-19

**Considerations for Tailoring State Machine Design Files for PLS and PROSE Devices**

| Device | Special Considerations |
|---|---|
| PLS | 1. The complement array is assigned as a fictional pin after the buried register nodes are defined in the pin list. Refer to *PLS Device General Considerations*, Section 4.3.1.<br><br>2. The complement array implements all state transition and output equation local defaults (specified by **+->**). The complement array is non-functional if all possible transitions are specified in the state transition equation. To use the array for other purposes, you must add the Equations segment (refer to *Build A Boolean Equation Design*, Chapter 4) and use Boolean equations.<br><br>3. The PLS devices have a pin you can configure as an output enable or preset pin. Configure the pin in the Defaults segment.<br><br>*To configure the pin as . . .* — *Specify . . .*<br>Output Enable — `OUTPUT_ENABLE`<br>Preset — `MASTER_RESET` (Default)<br><br>You can use MASTER_RESET to put the state machine into a known initial state. If OUTPUT_ENABLE is specified, your design must have an initial state. Without an initial state, the machine may not function as designed. Include an initial state in the state diagram so that you will remember to write the necessary state transition and output equations.<br><br>4. The polarity of a pin in the pin list and in the State segment must be the same, either both active-high or both active-low. |

Table 4-19 (Continued)

Considerations for Tailoring State Machine Design Files for PLS and PROSE Devices

| Device | Special Considerations |
|---|---|
| PLS (Continued) | 5. If you allow the software to automatically assign state bits, remember to assign NC to buried nodes or dual output/state bits. |
| PROSE | 1. The PMS14R21 device has a pin that is configured as an output enable or preset pin. Configure the pin in the Defaults segment.<br><br>*To configure the pin as . . .* — *Specify . . .*<br>Output Enable — `OUTPUT_ENABLE`<br>Preset — `MASTER_RESET` (Default)<br><br>MASTER_RESET may be used to put the state machine into a known initial state. If OUTPUT_ENABLE is specified, initialization may be built into the design. Without an initial state, the machine may not function as designed. Include an initial state in the state diagram so that you will remember to write the necessary state transition and output equations.<br><br>2. The software automatically assigns state bits.<br><br>3. The first state transition and output equations have a special syntax to place the machine in an initial state after power-up and the first clock: |

Table 4-19 (Continued)

Considerations for Tailoring State Machine Design Files for PLS and PROSE Devices

| Device | Special Considerations |
|---|---|
| PROSE (Continued) | ***Example***<br><br>Mealy:<br><br>`POWER_UP := VCC -> Starting_State_Name`<br>`POWER_UP.OUTF := VCC -> Starting_Outputs`<br><br>***Example***<br><br>Moore:<br><br>`POWER_UP := VCC -> Starting_State_Name`<br>`POWER_UP.OUTF := Starting_Outputs`<br><br>VCC is required for both Mealy and Moore machines.<br><br>4. The maximum number of transitions from a state is four. This limit includes the local or global default branch.<br><br>5. Parentheses, ( ), are not allowed in the Conditions segment. |

5.3.2

## PAL Device General Considerations

This section describes only the syntax and considerations that allow you to take advantage of special features on PAL devices. A discussion of the syntax and structure for building a state machine design begins in *Build A State Machine Design,* Section 5.2.

The following general considerations apply to designing state machines for PAL devices.

1. Do not use the default option OUTPUT_HOLD.

2. Use both EXPAND and MINIMIZE to process the design.

3. Explicitly build initialization into the design. Without initialization, the machine may not function as designed. Include initialization transitions in the state diagram so that you will remember to include the necessary state and output equations. Alternatively, if the device features set and reset functions, these functions may be used for initialization.

4. PAL16RA8, PAL20RA10, and PAL devices without registers do not support state machine designs.

Some PAL devices have programmable features that require adding an Equations segment and using Boolean equations after the State segment. *Functional Equations,* Section 4.1.3.4, discusses the structure and syntax for the set, reset, and three-state functions.

Table 4-20 lists the features and syntax considerations for specific PAL devices.

**Table 4-20**

**Special Features and Considerations for Specific PAL Devices**

| Device | Features | Considerations |
|---|---|---|
| PAL22RX8, PAL22V10 | Set, Reset, and Three-state | Require Boolean equations* |
| PAL32VX10 | Set and Reset | Require Boolean equations.* |
| | Internal Nodes | 1. The ten buried nodes must be assigned after VCC and the fictional 25th pin in the pin list. After the usual pin names are defined, you assign a name for each buried node beginning with node 14. The syntax for the buried nodes is the same as for the other pins.<br>2. State assignments are defined using the node names. For automatic state bit assignment, label the nodes to be assigned as NC. |

*The structure and syntax for Boolean equations is discussed in *Build A Boolean Equation Design,* Chapter 4. Also refer to *Tailor the Design For specific Devices,* Section 4.3.

Table 4-20 (Continued)

Special Features and Considerations for Specific PAL Devices

| Device | Features | Considerations |
|---|---|---|
| PAL32VX10 (Continued) | | 3. Output equations are defined using the node names listed after VCC and the fictional pin. Assigning nodes to output pins is done with Boolean equations.* |
| *The structure and syntax for Boolean equations is discussed in *Build A Boolean Equation Design,* Chapter 4. Also refer to *Tailor the Design For specific Devices,* Section 4.3. | | |

After your design is specified, you can proceed to *Build Simulation,* Chapter 6.

5.4

## Review A Simple Design

The state diagram and design in this section comprise a simple, complete design.

Figure 4-53 shows the state diagram for the 2-bit up/down counter from which the design in Figure 4-54 was built.

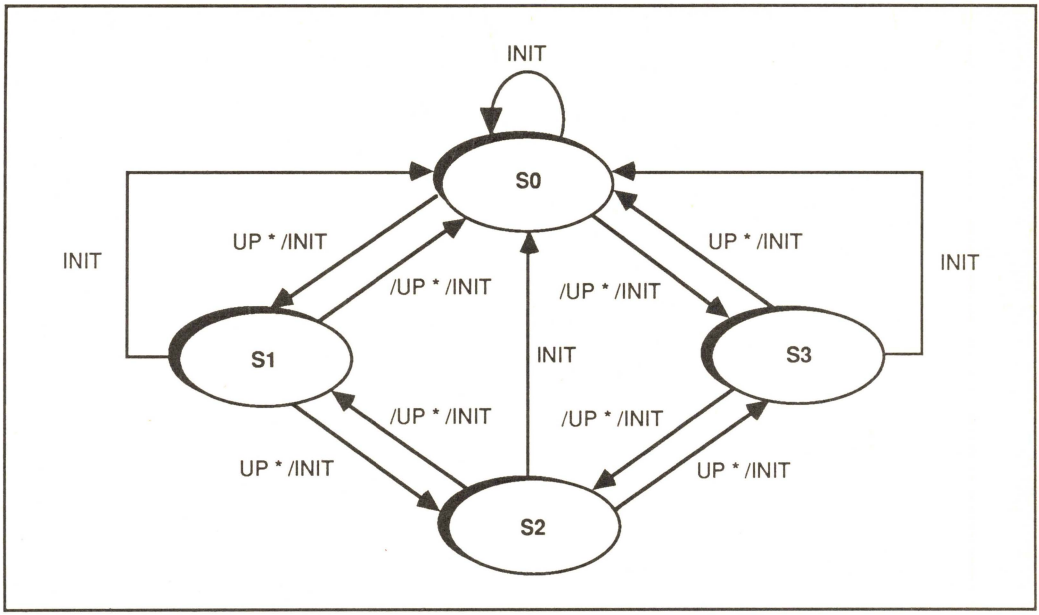



Figure 4-53

Simple State Diagram

```
TITLE           UP/DOWN COUNTER
PATTERN         X0000
REVISION        0
AUTHOR          BRYON MOYER
COMPANY         MONOLITHIC MEMORIES
DATE            9/23/87

CHIP    2_BIT_CTR       PAL16R4

;PINS  1    2     3    4    5    6    7    8  9    10
      CLK   UP  INIT   NC   NC   NC   NC   NC NC   GND

;PINS 11   12   13   14   15   16   17   18   19   20
     /OE   NC   NC   R0   R1   NC   NC   NC   NC   VCC

;THIS IS A SIMPLE TWO-BIT UP/DOWN COUNTER EXAMPLE. NOTE THAT
;INITIALIZATION HAS BEEN DESIGNED IN WITH DEFAULT_BRANCH.
;THE STATE MACHINE RETURNS TO STATE S0 IF CLOCKED WHEN PIN
;INIT IS HI. COUNTING IS ONLY POSSIBLE IF INIT IS LO.  IF
;PIN UP IS HI, COUNTER COUNTS UP.  IF PIN UP IS LO, COUNTER
;COUNTS DOWN.

STATE                       ;START THE STATE MACHINE SECTION

MOORE_MACHINE
DEFAULT_BRANCH S0           ;FOR INITIALIZATION

;IN THIS EXAMPLE, THE OUTPUTS ARE TAKEN DIRECTLY FROM THE
;STATE BITS.  THEREFORE, AN OUTPUT DEFAULT IS NOT USED.
```

Figure 4-54 (One of Two)

Design for Figure 4-53

```
;STATE ASSIGNMENTS
;STATES ARE THE SAME AS THE OUTPUTS

S0 = /R1 * /R0   ;COUNT 0
S1 = /R1 *  R0   ;COUNT 1
S2 =  R1 * /R0   ;COUNT 2
S3 =  R1 *  R0   ;COUNT 3

;STATE AND TRANSITION DEFINITIONS
;BECAUSE THE OUTPUTS ARE TAKEN DIRECTLY FROM THE STATE BITS,
;NO OUTPUT EQUATIONS ARE USED.


S0 := COUNT_UP           -> S1          ;COUNT 0 TO 1
   +  COUNT_DOWN         -> S3          ;COUNT 0 TO 3

S1 := COUNT_UP           -> S2          ;COUNT 1 TO 2
   +  COUNT_DOWN         -> S0          ;COUNT 1 TO 0

S2 := COUNT_UP           -> S3          ;COUNT 2 TO 3
   +  COUNT_DOWN         -> S1          ;COUNT 2 TO 1

S3 := COUNT_UP           -> S0          ;COUNT 3 TO 0
   +  COUNT_DOWN         -> S2          ;COUNT 3 TO 2

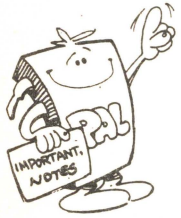

;DEFINE THE BRANCH CONDITIONS.  THE BRANCH CONDITIONS ARE
;ESSENTIALLY THE CONDITIONS FOR COUNTING UP OR DOWN. THE
;COUNTER CAN ONLY COUNT IF PIN INIT IS LO.  IF PIN INIT IS
;HI, THEN NEITHER OF THE CONDITIONS IS TRUE, AND THE DEFAULT
;BRANCH IS USED TO INITIALIZE THE COUNTER TO COUNT 0.

CONDITIONS

COUNT_UP =        UP*/INIT
COUNT_DOWN =     /UP*/INIT
```

Figure 4-54 (Two of Two)

Design for Figure 4-53


IMPORTANT
NOTES

# 6. Build Simulation

## *About This Chapter*

This chapter describes the additional commands and control structures that allow you to simulate your design. Though simulating is optional, verifying a design before programming a device saves time. Simulating the design helps verify that the equations do implement the required function.

## *About PALASM 2 Software And Simulation*

Simulating the design means specifying a trial set of input values for your design and checking that the resulting outputs are correct.

PALASM 2 software has an event-driven simulator supporting all PAL device architectures, both asynchronous and synchronous. The program realistically simulates events generated by asynchronous or synchronous feedback and external events you generate. The simulator detects and reports oscillatory conditions and conflicts in the expected and the actual values of any signal.

| *To...* | *Refer to Section...* |
|---|---|
| Review the special syntax rules | 6.1 |
| Build the Simulation segment | 6.2 |
| Review a sample design and interpret the output files | 6.3 |

6.1

## Special Syntax

The general syntax rules for Boolean and state machine designs apply to building the Simulation segment except for the characters in Table 4-21.

**Table 4-21**

**Exceptions to General Syntax**

| Character(s) | Function |
|---|---|
| < | Less than operator |
| > | Greater than operator |
| = | Equality operator |
| <= | Less than or equal to . . . |
| >= | Greater than or equal to . . . |

6.2

## Build The Simulation Segment

Information in the Simulation segment defines a trial set of inputs and tells the software what to do with them. This segment is the last part of the design as shown in Figure 4-55.

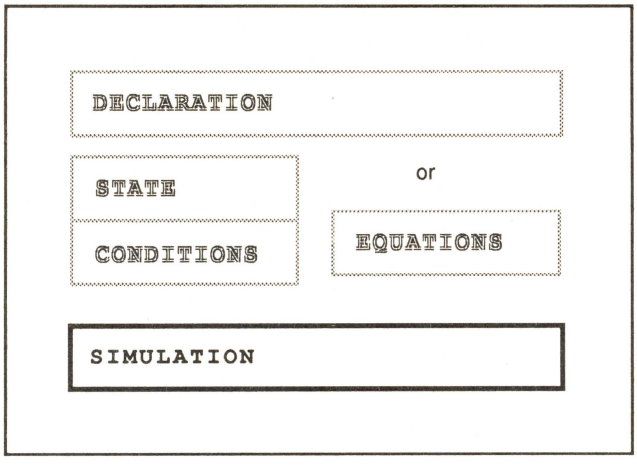



Figure 4-55

Location of the Simulation Segment

The keyword

SIMULATION

is required to identify this segment of the design.



6.2.1

## The Simulation Language

The simulation language has English-like words, making it easy to read and understand. It offers iterative looping, conditional branching, setting of signals, verification of signal values, and selective observation of signals. Table 4-22 briefly describes each directive in the simulation language.

Table 4-22

Description of Simulation Commands

| Command | Description |
|---|---|
| PRLDF | Initializes register outputs on preloadable devices |
| SETF | Specifies new input values |
| CLOCKF | Generates a clock signal on the dedicated clock pin |
| CHECK | Verifies that the expected values and the simulated values are the same |
| TRACE_ON | Defines specific signals to record in a special output file |
| TRACE_OFF | Turns off the TRACE_ON command. |
| FOR . . . TO . . . DO loop | Iterates a set of commands a fixed number of times |
| WHILE . . . DO loop | Iterates a set of commands until a condition is satisfied |
| IF . . . THEN . . . ELSE | Conditional branching |

After running simulation with assembling first, the software stores results in two output files: a history file (FILENAME.HST) and a trace file (FILENAME.TRF). The history file contains the values of all signals from the start of simulation to the end. The trace file contains the values of the signals mentioned between TRACE_ONand TRACE_OFF commands.

The organization of directives and the size of the segment depend on how thoroughly you simulate the design. You may need to use all the directives discussed below, or use only a few.

If your design uses feedback from output registers and the device has a preload pin, you can preload register outputs to initialize the registers before initializing inputs (refer to *PRLDF,* Section 6.2.1.1). However, preloading registers after initializing the clock and all inputs is a better approach (refer to *SETF,* Section 6.2.1.2).

6.2.1.1

## PRLDF

The PRLDF command assigns logical values to, or initializes, register outputs on preloadable PAL devices:

***Syntax***

PRLDF List of registered output pins

***Example***

```
PRLDF 01 /02 /03
```

The elements of PRLDF are registered output pin names. Uncomplemented names cause a high logic value to be assigned to the registered output pins; names preceded by / assign a low logic value.

This command affects the flow of simulation differently according to the way each registered device has its preload configured:

- For devices that have a dedicated preload pin, PRLDF successively disables the outputs, enables preload, loads the registers with the required logic values, disables preload, and finally enables the outputs.
- For devices that have their registers preloaded with supervoltages, PRLDF places a P in the clock field of the JEDEC vector. Simulation continues with the new value.
- For registered devices that cannot be preloaded, PRLDF provides a convenient way of initializing registers to desired values.

Remember the following guidelines for using PRLDF.

1. PRLDF the PAL32R16 and PAL64R32 in banks of eight.
2. Only registered output pin names are valid arguments to the PRLDF command.
3. The registers are preloaded, not the output pins. The output pins carry the results.
4. When PRLDF is used on a state machine, both the state and its outputs must be preloaded, unless the state bits and the output bits are the same:

***Example***

```
PRLDF STATE0 /OUT1 OUT2 OUT3 /OUT4
```

5. On certain PAL devices, such as the PAL20X4, the A version of the part preloads with supervoltages, while the standard version does not. In these cases, the software preloads the device and issues a warning message to you.

With your registers preloaded, you can initialize the clock and inputs as described next.

6.2.1.2

## SETF

SETF specifies new input values for the software to simulate. SETF is usually the first command after PRLDF when simulating registered output devices:

***Syntax***

SETF List of input pins

***Example***

```
SETF A /OE B /RESET /D0 D1 D2
```

At the start of simulation, all signals are undefined and show an X in the simulation output. A signal should only be set if you want to change it from the previous value. The signals in SETF are set high (H in the output) if not preceded by / ; otherwise, they are set low (L in the output). In the above example A, B, D1, and D2 are all set to H; OE, RESET, and D0 are all set to L. Other input signals are undefined (X) or remain at a previously defined level.

When a SETF command affects outputs, a vector is generated and all the equations that are affected are evaluated. Internally generated events are also detected and evaluated. With some activities, many more vectors can be generated by a single SETF command than with others because of feedback and asynchronous events. Although you do not see it, the simulator continues generating vectors and evaluating equations until the system stabilizes; that is, until there are no more changes in the output signals or no more events are generated. If the system fails to stabilize after ten iterations, then an oscillatory condition is assumed, and the simulation halts.

After initializing the registers, the clock, and the inputs, you may want to verify the circuit's operation after a clock pulse for registered output designs.

## Controlling Output Enable With SETF

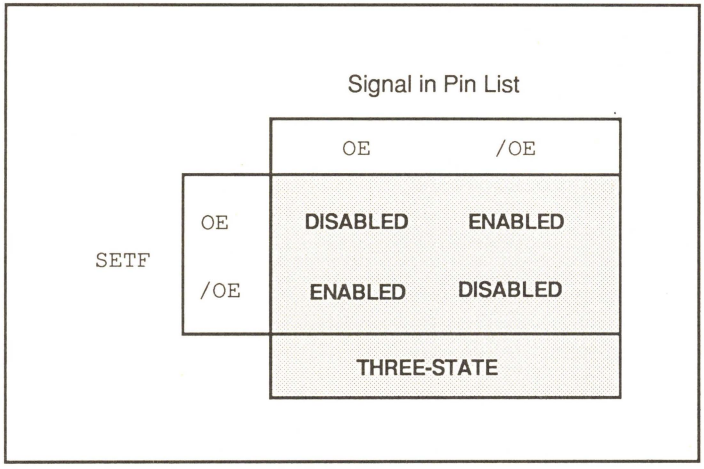
1. You have two options for using SETF to control the three-state function in simulation:

   **Option 1:**

   Table 4-23 helps you determine how to use SETF and what consequences to expect from a device with a dedicated output enable pin (OE). Enabled means visible output (three-state buffer is high); disabled means no output (three-state buffer is low).

**Table 4-23**

**Table for Using SETF to Control a Dedicated Output Enable**




In Table 4-23,

| *If the pin is defined as...* | *And you want the outputs...* | *Enter...* |
|---|---|---|
| OE | Disabled | SETF OE |
| OE | Enabled | SETF /OE |
| /OE | Disabled | SETF /OE |
| /OE | Enabled | SETF OE |

**Option 2:**

If a local product term controls the three-state (such as for PAL22V10 and PAL20RA10), where

Output_Pin.TRST = Enable_Condition

then

```
SETF ENABLE_CONDITION
```
(Enable_Condition is a logic 1; three-state is enabled)

```
SETF /ENABLE_CONDITION
```
(/Enable_Condition is logic 0; three-state is disabled)

2. To use a PAL10H20G8 I/O pin as an input, set the sum of product terms and the output pin to low in the Simulation segment:

***Example***

```
EQUATIONS

   OUT1 = A * C + B * D

SIMULATION

   SETF /A /B
   SETF OUT1      ;use as an input
```

6.2.1.3

### CLOCKF

CLOCKF generates a clock signal on the dedicated clock pin(s):

***Syntax***

CLOCKF list of clock signal(s)

***Example***

```
CLOCKF CLK1
```

CLOCKF specifies the clock signals (dedicated clock pins) to which the software applies a clock pulse. Only the clock pin(s) of the device can be used in the CLOCKF command; any other pin is an illegal signal for this command.

Using SETF, initialize the clock pin to low in the first line of the Simulation segment before using CLOCKF:

***Example***

```
SETF /CLK
PRLDF OUT1
CLOCKF CLK
```

if the pin was defined as CLK in the pin list. If the clock pin has a high value at the first CLOCKF command, an error occurs. Notice that you can control the clock pin using SETF:

Each CLOCKF command corresponds to a pulse going from low to high to low. Thus, two or three vectors are generated. During the positive edge transition, the new value of the registers being clocked is transferred to the output. No action takes place for the registers that are not clocked.

At every CLOCKF command, internally generated events and asynchronous events are detected. More vectors are generated until the circuit stabilizes. The operation of CLOCKF is similar to the SETF command, except that CLOCKF generates a pulse (return-to-zero) rather than maintaining a level (non-return-to-zero).

**Note**: On the PAL10H20GR8, do not CLOCKF pin 3 if pin 3 serves as an input for combinatorial equations and as clock/latch enable for registered/latched equations. Use two SETF lines to imitate the low-high action of CLOCKF.

***Example***

```
SETF /CLK
SETF CLK
```



6.2.1.4

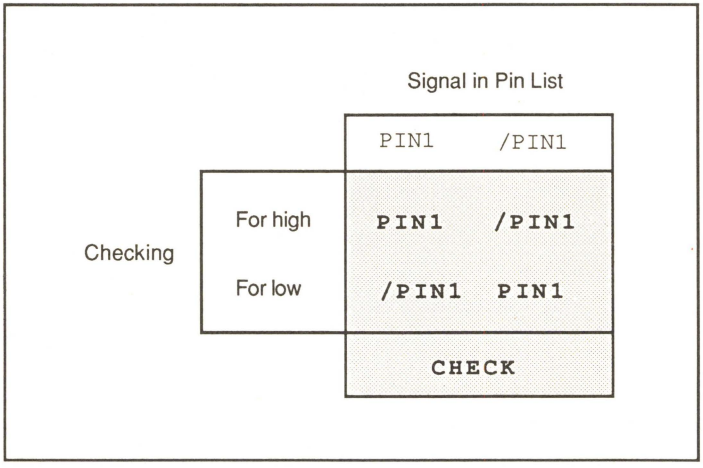
### CHECK

CHECK can be used at important points in your simulation for debugging and verifying your design. CHECK verifies that the the signals you expect actually occur. If the signals you expect differ from the simulated signals, the software reports an error. How you specify the signals to check depends on the polarity of the signal in the pin list.

Table 4-24 summarizes how to CHECK signals.

Table 4-24

Table for Checking Signals




In Table 4-24,

| *If the pin is defined as...* | *And you are checking for a...* | *Enter...* |
|---|---|---|
| PIN1 | High | CHECK PIN1 |
| PIN1 | Low | CHECK /PIN1 |
| /PIN1 | High | CHECK /PIN1 |
| /PIN1 | Low | CHECK PIN1 |

***Syntax***

CHECK List of signals you expect

***Example***

```
CHECK Q0 /Q1 /Q2
```

Only outputs can be arguments to the CHECK statement. Whenever a CHECK is executed, the simulator compares the actual value and the expected value of a particular signal. If they are equal, no action is taken. If they are not equal, the simulator reports a warning and continues processing using the actual value. CHECK reports the warning by

placing a **?** in the vector where the error occurred as well as a vector number. The simulation output files contain the **?** at this particular location.

6.2.1.5

## TRACE_ON

TRACE_ON defines specific signal values to record in the simulation trace file (FILENAME.TRF). By specifying only the signals significant to you, you can more easily read the simulation trace results:

***Syntax***

TRACE_ON list of input and/or output signals

***Example***

```
TRACE_ON /OE SET RESET D0 D1 D2 D3 /Q0 /Q1 /Q2
```

This command contains the signals that you want listed in the trace file. The signal names will be listed in the same order and with the same polarity as present in the TRACE_ON command. This list of signals will be active until the next TRACE_OFF command or until the end of the simulation specification. New signals can be traced on after the TRACE_OFF command. This command helps you group the signals more naturally for debugging purposes. For example, all control signals can be grouped together, then all data signals can be grouped together, and then all output signals can be grouped together. This makes observing the results in the trace file very easy.



6.2.1.6

## TRACE_OFF

TRACE_OFF turns off the TRACE_ON command. The signals you were tracing will no longer be recorded in the trace file. After this command, no more results are added to the trace file until the next TRACE_ON command is given.

Remember that the history file (FILENAME.HST) contains all the information generated from the start of simulation to the end. The signals are in the same order and of the same polarity as in the pin list. The trace options break your results into time frames, which is critical for debugging. You can make the signals appear in any order with any polarity in the trace file.

6.2.1.7

## FOR . . . TO . . . DO

The FOR loop allows repetitive execution of statements:

***Syntax***

```
FOR index var := lower limit TO upper limit DO
    BEGIN
        statements
    END
```

***Example***

```
FOR J := 3 TO 8 DO
    BEGIN
        SETF A /B
        CLOCKF CLK
    END
```

Many statements can be embedded in a FOR loop, including another FOR construct with a different indexing variable. You can generate many vectors just by increasing the limits of this loop. The lower limit should be less than or equal to the upper limit. All the limit values should be greater than or equal to zero. You cannot use negative values for the limits. The loop is not executed if the conditions expressed in the limits are equal.

6.2.1.8

## WHILE . . . DO

The WHILE loop allows a repetitive execution of statements that may be controlled by evaluation of logic conditions present within the device:

***Syntax***

```
WHILE condition DO
    BEGIN
        statements
    END
```

Many statements can be embedded in a WHILE loop, including other looping constructs. The WHILE loop is used to iterate a set of commands until the condition is false.

Condition expressions cannot contain nested parentheses. The condition can be any Boolean expression of logic signals or mathematical equality ( **=**, **>**, **<**, **>=**, **<=**, **<>** ):

***Example***

```
WHILE (I<2) DO
```

Here, the simulator checks if the condition I < 2 is true. The condition can also be any Boolean expression:

***Example***

```
WHILE (DRDY * /CLR) DO
```

Here, the simulator evaluates (DRDY * /CLR). If it is true, then the condition is true.

6.2.1.9

## IF . . . THEN . . . ELSE

Use this construct for conditional branching:

***Syntax***

```
IF condition THEN                or   IF condition THEN
    BEGIN                                 BEGIN
        statements                            statements
    END                                   END
ELSE
    BEGIN
        statements
    END
```

***Example***

```
IF J = 5 THEN
   BEGIN
      CHECK Q0
   END
ELSE
   BEGIN
      CHECK /Q0
   END
```

The two ways to use this construct are: with an ELSE clause or without. If the construct has an ELSE clause and the condition is true, the THEN clause is executed; otherwise, the ELSE clause is executed. If there is no ELSE clause and the condition is not true, then the simulation executes the next command or construct after the IF . . . THEN construct.

As with the WHILE . . . DO construct, condition expressions cannot contain nested parentheses. The condition can be any Boolean expression of logic signals or mathematical equality ( =, >, <, >=, <=, <> ).

6.2.2

## Review Simulation Guidelines

1. All signals are assumed to be don't care at the start.

2. Initialize all your control signals (such as three-state, preload and clock) to their default values. If they are not initialized, the simulation may give erroneous results and may generate warnings for pins not initialized.

3. If the three-state or preload pin is /OE, for example, then SETF OE will enable the outputs and SETF /OE will disable the outputs.

4. For the PAL20RA10:

   If A, B, CLK, RESET, and SET are defined in the pin list and

```
Q0 := A * B
Q0.CLKF = CLK
Q0.RSTF = RESET
Q0.SETF = SET
```

   appear in the Equations segment, then the following simulation commands will have the given results:

```
SETF  SET /RESET     ;The register Q0 is set to H,
                     ;so the output pin will go L.
SETF  RESET /SET     ;The register Q0 is set to L,
                     ;so the output pin will go H.
```

   The data path of this device is treated in the normal way because the polarity fuse is in front of the register. The simulator takes care of any difference in the polarity between the signal in the pin list and the left side of the equation.

6.2.3

## Rules For State Machine Simulation Syntax

1. In a Boolean equation design, you use the PRLDF, CHECK, TRACE_ON, WHILE and IF constructs to reference the value of an output. In a state machine design, however, you use these constructs to reference states *and* outputs unless state bits and output bits are the same:

   ***Example***

   ```
   PRLDF STATE_ONE 01 /02 /03
   ```

2. The two history output files from simulation are FILENAME.TRF (if TRACE_ON is used) and FILENAME.HST. In addition to the H, L, X, and Z (high-impedance) values for signals, output files for the PMS14R21 contain the state of the machine at each point in the simulation.

6.3

## Review A Sample Design And Interpret The Output Files

The following sections discuss interpreting the simulation output files for SUPER.PDS in Figure 4-56. *Run The Software,* Chapter 3, discusses processing SUPER.PDS. Along with the previous discussions on simulation commands, the following discussions help you understand how to simulate a design and interpret the results in waveform and output file formats.

```
TITLE      SUPER_FRAME_PAL
PATTERN    SUPER FRAME PAL FOR T1 INTERFACE
REVISION   P1.02
AUTHOR     STEVE PATTERSON AND THERESA SHAFER
COMPANY    MONOLITHIC MEMORIES
DATE       1/16/87

;DESCRIPTION

;This PAL counts the T1 Frames and controls the Signal
;Bits extraction process, including Fly Wheeling.  It
;also provides various other signals which indicate
;the frames with signal bits.  The counter is reset
;with either RSTB or when frame detection is SUNK and
;frame 1 occurs from two different sources (FRM1 & SOF).

CHIP SUPER_FRAME PAL16R6

;PINS
;1      2    3      4      5    6  7  8  9  10
T1_CKB RSTB FRM1B SUNK   SOF NC NC NC NC GND

;11 12 13 14 15 16 17     18      19 20
OEB NC Q3 Q2 Q1 Q0 FRM_6 FRM_12 NC VCC
```

Figure 4-56 (One of Three)

SUPER.TRF Input File

```
;INPUTS:T1_CKB ACTIVE LOW EXTERNAL T1 CLOCK
;RSTB  ACTIVE LOW MASTER RESET
;SOF   LAST KNOWN START OF FRAME
;SUNK    ACTIVE HIGH SIGNAL INDICATING "IN FRAME SYNC"
;OEB   ACTIVE LOW OUTPUT ENABLE INPUT

EQUATIONS

/Q3 := /Q2 * Q1 * Q0
    + /Q3 * /Q2
    + /Q3 * /Q1
    + /Q3 * /Q0
    + /FRM1B * SOF * SUNK
    + /RSTB

/Q2 := Q2 * Q1 * Q0
     + /Q2 * Q3
     + /Q2 * /Q1
     + /Q2 * /Q0
     + /FRM1B * SOF * SUNK
     + /RSTB

/Q1 := Q1 * Q0
     + /Q1 * /Q0
     + /FRM1B * SOF * SUNK
     + /RSTB

/Q0 := Q0
    + /FRM1B * SOF * SUNK
    + /RSTB

/FRM_6  := Q3
         + /Q2
         + Q1
         + Q0

/FRM_12 := /Q3
         +  Q2
         + /Q1
         + Q0
```

Figure 4-56 (Two of Three)

SUPER.PDS Input File

```
;OUTPUTS:Q(3-0) STATE VARIABLES
;FRM_6  CLOCK SIGNAL WHICH INDICATES SIGNAL BIT A
;FRM_12 CLOCK SIGNAL WHICH INDICATES SIGNAL BIT B

SIMULATION

TRACE_ON T1_CKB RSTB FRM1B SOF SUNK
Q3 Q2 Q1 Q0 FRM_6 FRM_12

SETF /OEB      ; ENABLE OUTPUT
     /RSTB     ; RESET REGISTERS
     /T1_CKB   ; INITIALIZE CLOCK
CLOCKF T1_CKB
SETF RSTB /SOF FRM1B SUNK
CLOCKF T1_CKB

FOR I:=1 TO 24 DO
 BEGIN
 CLOCKF T1_CKB
 END

SETF /SUNK SOF /FRM1B
CLOCKF T1_CKB
SETF /SUNK /SOF /FRM1B
CLOCKF T1_CKB
SETF /SUNK SOF FRM1B
CLOCKF T1_CKB
SETF /SUNK /SOF FRM1B
CLOCKF T1_CKB
SETF SUNK SOF /FRM1B
CLOCKF T1_CKB
SETF SUNK /SOF /FRM1B
CLOCKF T1_CKB
SETF SUNK /SOF FRM1B
CLOCKF T1_CKB
SETF SUNK SOF FRM1B
CLOCKF T1_CKB
TRACE_OFF
```

Figure 4-56 (Three of Three)

SUPER.PDS Input File

After processing, the simulation results are stored in the following output files:

- History file (FILENAME.HST)
- Trace file (if TRACE_ON used, FILENAME.TRF)

**Note**: The simulation program also creates a JEDEC fuse and test data file (FILENAME.JDC) if assembly was done first. This file can be downloaded to the programmer to program and verify the device. For more information about this file, refer to *Interpret the JEDEC Test Data*, Section 6.3.6, and to *Program The Device,* Chapter 7.

You may view the files as waveforms on your screen or on a printout as discussed in *View The Simulation Output Files,* Section 3.7.2.

Figure 4-57 shows the format of the waveform displays and output files. In addition to the previous discussions of the simulation commands, the following sections help you interpret the codes in simulation waveforms and output files.

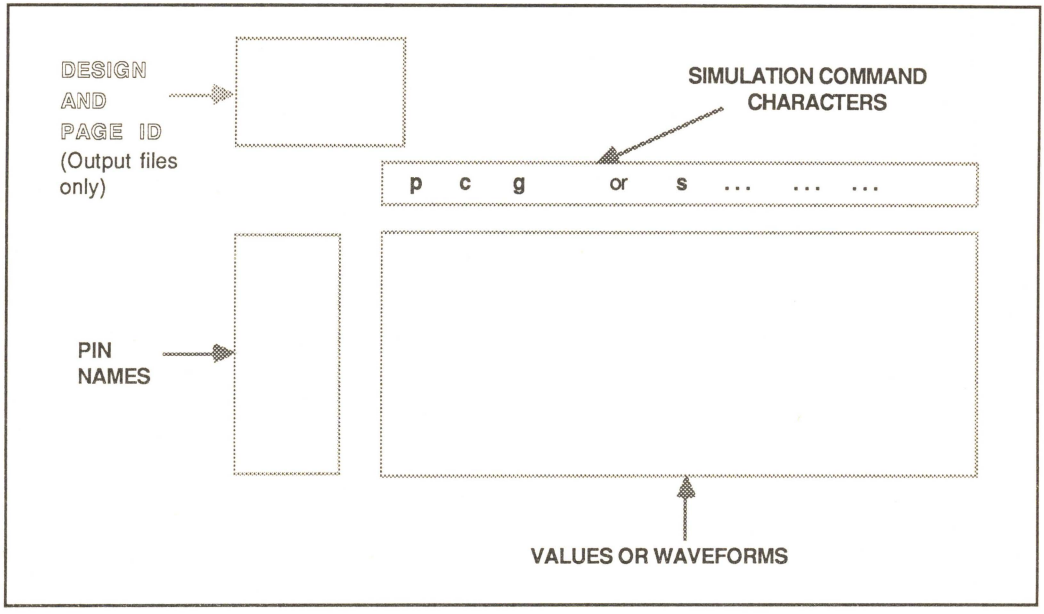



**Figure 4-57**

**Waveform Display and Output File Format**

Table 4-25 shows the characters and their corresponding simulation commands.

**Table 4-25**

**Simulation Output Characters**

| Character | Input File Command |
|---|---|
| g (s on a PROSE device) | SETF |
| c | CLOCKF |
| p | PRLDF |

In addition to the previous discussions on simulation commands and constructs, the following discussions help you simulate your design and interpret the results.

## The g Character

The character g (s on a PROSE device) indicates the SETF command. In the Simulation segment, you specify the pins that are set to high or low values with this command. When the value on a pin does not change, the result is not recorded in the history file. For example, if a pin that has a high value already, is set high again, the second high value causes no change and generates no extra vectors. The values under the g character can take up more than one column. When this happens, only the last column indicates the stable values. The first two columns are caused by intermediate conditions such as feedbacks.

**Note**: A column of SETF values can also be caused by a PRLDF command in the input file. When this occurs, the g column will immediately follow the p column.

## The c Character

The c character indicates the CLOCKF command. Notice in Figure 4-58 that each column with a c character is preceded by two more columns of values. This is because the clocking procedure consists of three steps:

1. Raise the clock pin. The clock pin goes from low to high.

2. While the clock pin is high, the new output pin values are recorded.

3. Lower the clock pin. The results are shown in the last column.

### The p Character

The character p appears in the history output when a PRLDF or preload command is specified in the Simulation segment. This command causes a three-step procedure on a registered device:

1. The output enable pin is set high.

2. A value is loaded in the register.

3. The output enable pin is set low.

The history file records the entire preload procedure. The column headed by the p character records the values. However, there are two exceptions:

- The outputs may be stored in the next column which is headed by a g or s. This occurs when the value in the register changes.

- If the PRLDF command is followed by a CLOCKF in the input file, the first CLOCKF column records the stable values of the preload.

6.3.1

## Interpret The History Waveforms

History waveforms are based on the information in the history file. The waveforms are not stored in a separate file; they graphically represent the FILENAME.HST file. Figure 4-58 shows SUPER.HST as history waveforms.

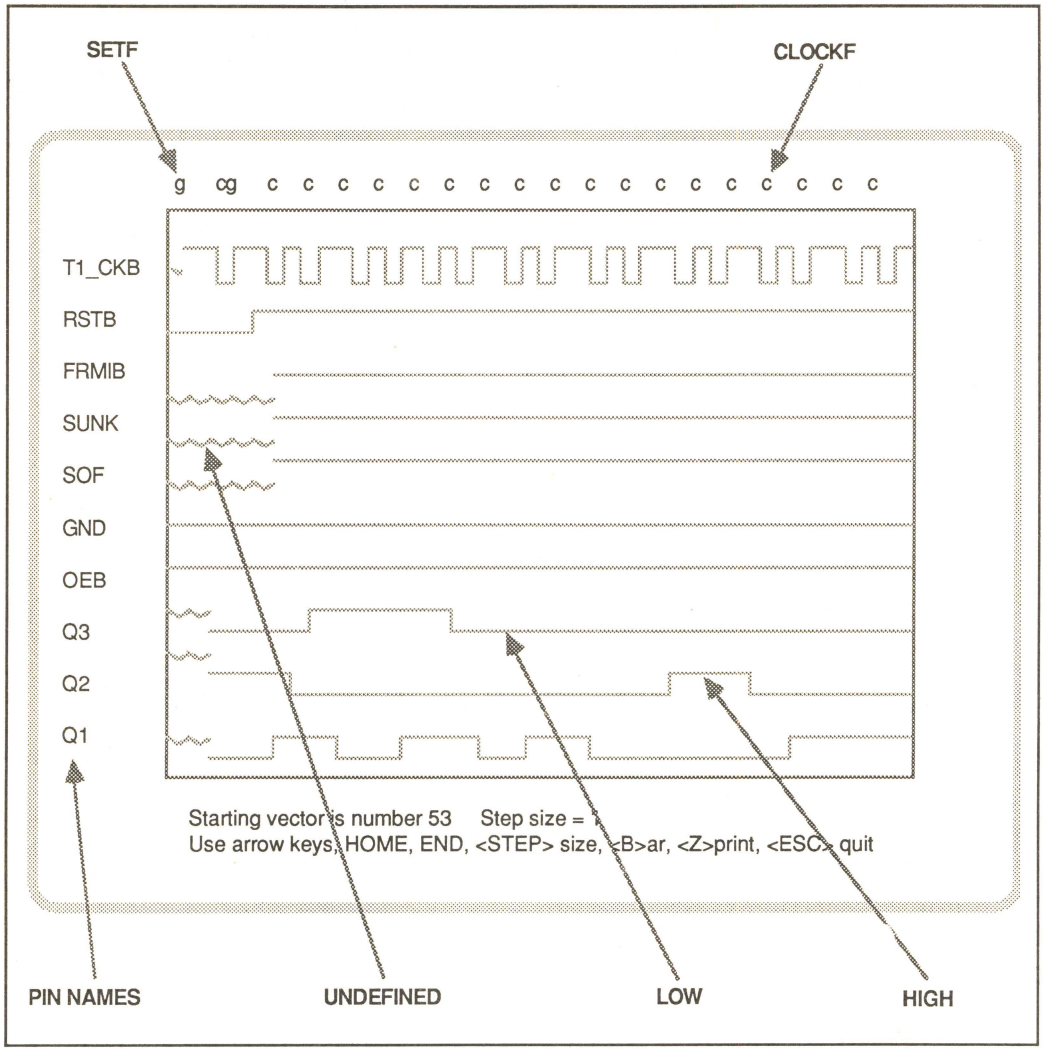



Figure 4-58

SUPER.HST Waveforms

You read waveforms in columns. Use the vertical bar cursor to track the events in each column.

The SETF and CLOCKF commands used in the Simulation segment of the input file are coded in the horizontal row above the waveforms. By moving the bar across the screen you can track the results caused by the SETF and CLOCKF commands on each pin. Notice that the p character does not appear. This is because this example does not contain a PRLDF command.

The pins for which simulation events were defined appear vertically at the left of the waveforms.

**Note**: The history waveforms list the pins exactly as they are defined in the pin list of the input file. Therefore, they are in the same sequence and have the same polarity as the pin list. NC or no-connect pins are not included. The signals in the history waveform can also be observed on the pins of the device.

6.3.2

## Interpret The Trace Waveforms

The trace waveforms are based on the information in the trace file. The waveforms are not stored in a separate file; they graphically represent the FILENAME.TRF file. Figure 4-59 shows SUPER.TRF as trace waveforms.

The events recorded by trace waveforms are determined by the TRACE_ON and TRACE_OFF commands in the Simulation segment. These commands allow you to trace a group of signals in the order and polarity you specify.

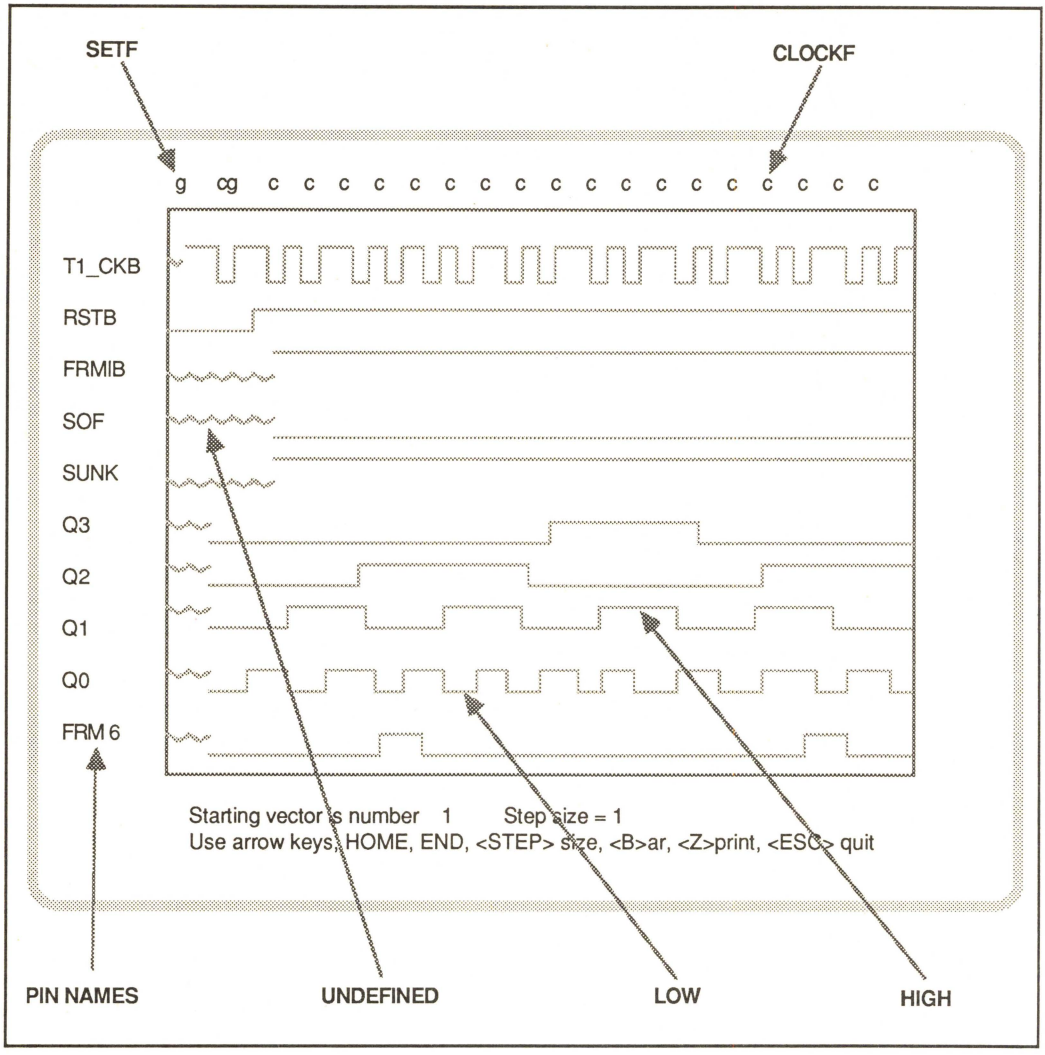



Figure 4-59

SUPER.TRF Waveforms

Track the events in columns using the vertical bar cursor just as you tracked the history waveforms.

The trace waveforms differ from the history waveforms in the following ways:

- The signal names in the history waveforms are taken from the pin list; the signal names in the trace waveforms are taken from the TRACE_ON command. Therefore their polarity may be inverted.
- The signal in the trace file may be in any order.
- Some signals or events may not be displayed. The trace waveforms show only those signals between the TRACE_ON and TRACE_OFF commands; the history waveforms are more complete.

Interpreting the history and trace files is discussed next.

6.3.3

## Interpret The History File

The history file contains simulation results in columns. If you prefer viewing the simulation results in waveforms, you may not need to look at the sample history file in Figure 4-60.

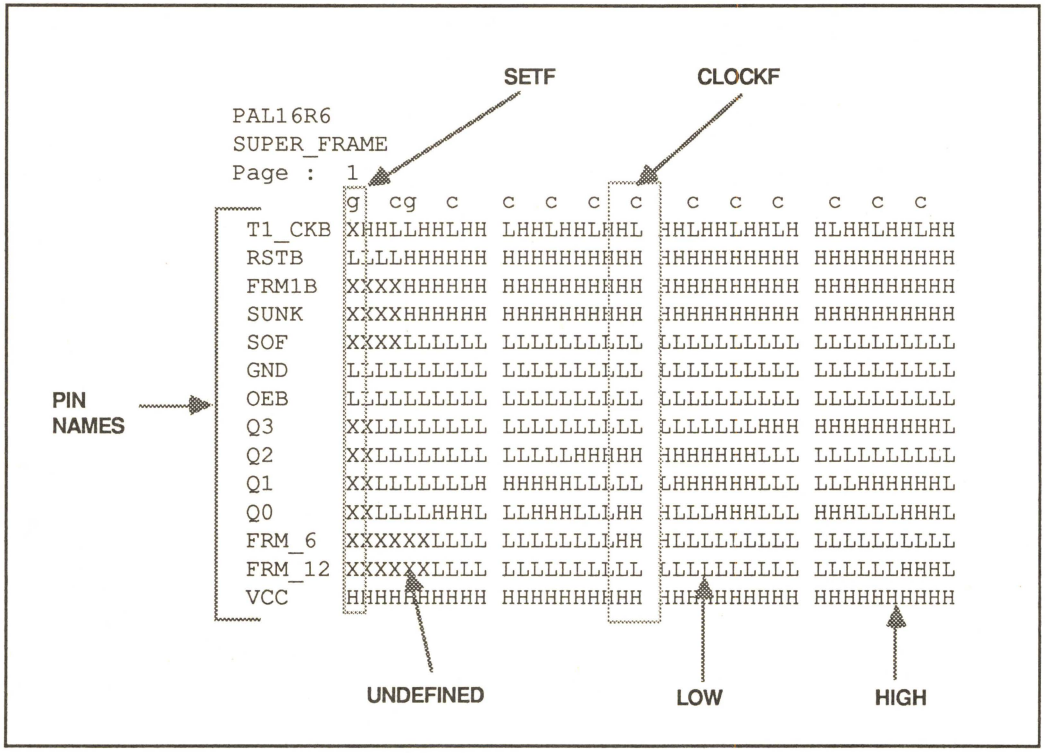



Figure 4-60

SUPER.HST File

Notice that you read the file in columns. On the left side of the page, the pins are listed in order beginning with pin 1. The characters for the simulation commands are displayed in the first horizontal row on each page of the file. For example, the values in the outlined columns in Figure 4-60 are a result of SETF (g) and CLOCKF (c) commands. Refer to Table 4-26 for definitions of the command characters.

Each column contains the values on each pin that result from simulation commands. Table 4-26 lists the values that appear in a history file.

Table 4-26

Value Characters in the History File

| Character | Value |
|---|---|
| H | High |
| L | Low |
| Z | High impedance state |
| X | Undefined or don't care value |
| ? | CHECK command discrepancy: simulated value does not match the expected value |

**Note**: The example in Figure 4-60 does not contain p, ? or z characters. This is because the input file does not contain PRLDF statements, CHECK statements, or high impedance states.

**Note**: The history file lists the pins exactly as they are defined in the pin list. Refer to the note in *Interpret The History Waveforms,* Section 6.3.1.



6.3.4

## Interpret A PROSE History File

The state machine history file for a PROSE device differs from a standard history file in one respect. It shows the state the machine is in at the bottom of the file. Figure 4-61 displays the state information at the bottom of a PMS14R21 history file. Notice how the values of the states can be tracked vertically against each pin.

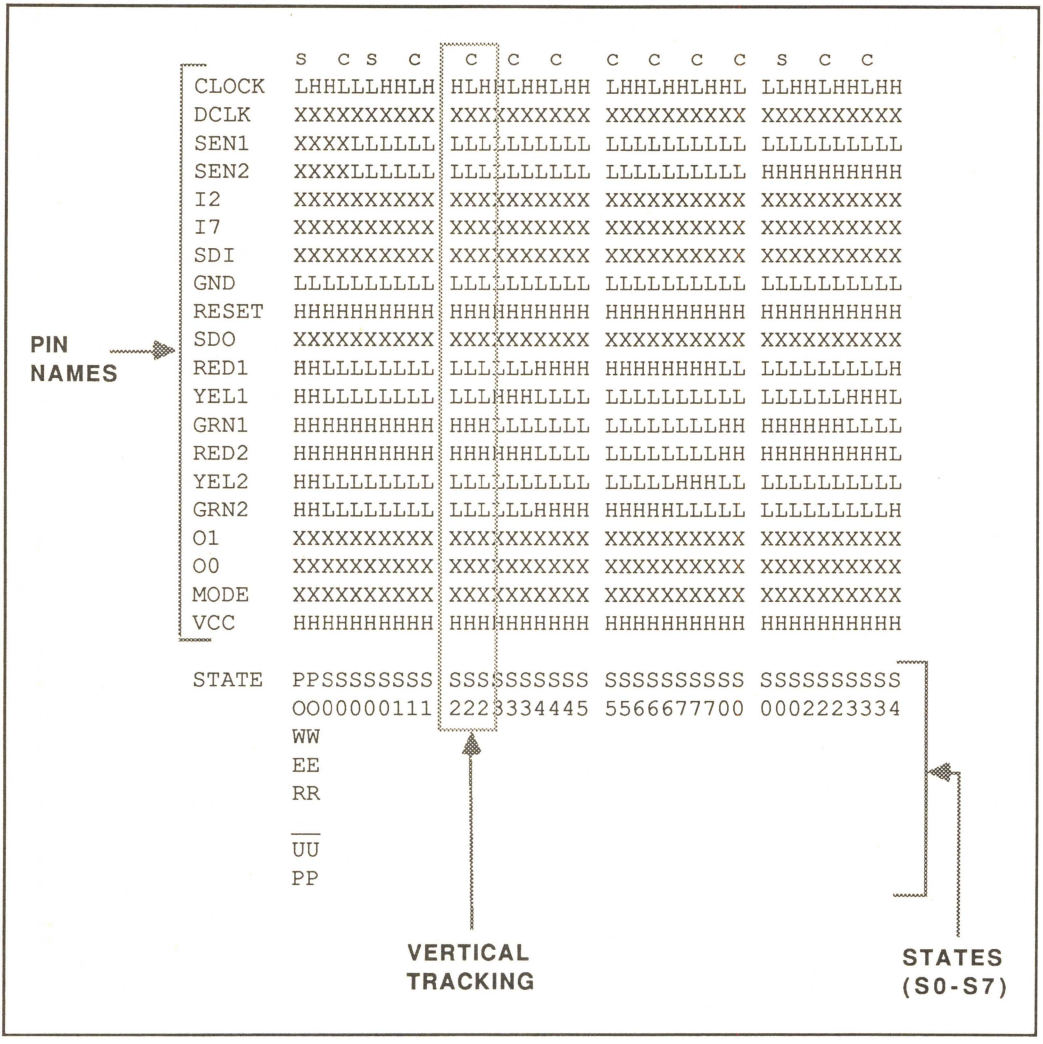



Figure 4-61

Sample PMS14R21 History File

6.3.5

## Interpret The Trace File

The trace file is a result of the TRACE_ON command in the input file and is read in the same way as the history file. As mentioned earlier in the discussion on waveforms, the trace file differs from the history file in the following ways.

- The pin names in the trace file are taken from the TRACE_ON command, not from the pin list. Therefore, their polarity may be inverted.
- The pins in the trace file may be in a different order.
- Some pins or events may not be displayed. The trace file traces only those pins you define in the TRACE_ON command; the history file is more complete.

Figure 4-62 shows a sample trace file. Notice that the GND, VCC, and OEB pins shown in Figure 4-61 are not listed.

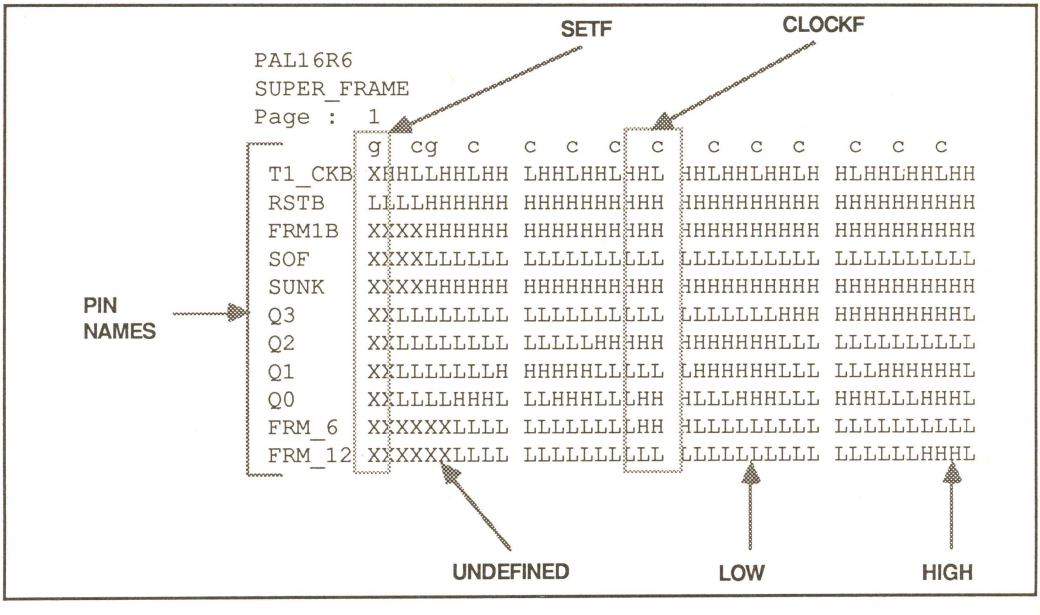



Figure 4-62

Sample Trace File

6.3.6

## Interpret The JEDEC Test Data

The simulation program also generates a JEDEC test data file that has the extension .JDC if you assembled the design first. This file contains the fuse data with test vectors at the bottom. The device programmer can use this file for programming and verifying the device. Refer to *Program The Device,* Chapter 7, for more information.

Figure 4-63 displays the test vectors from the SUPER.JDC output file. Notice that the figure does not include the fuse map that appears above the test vectors.

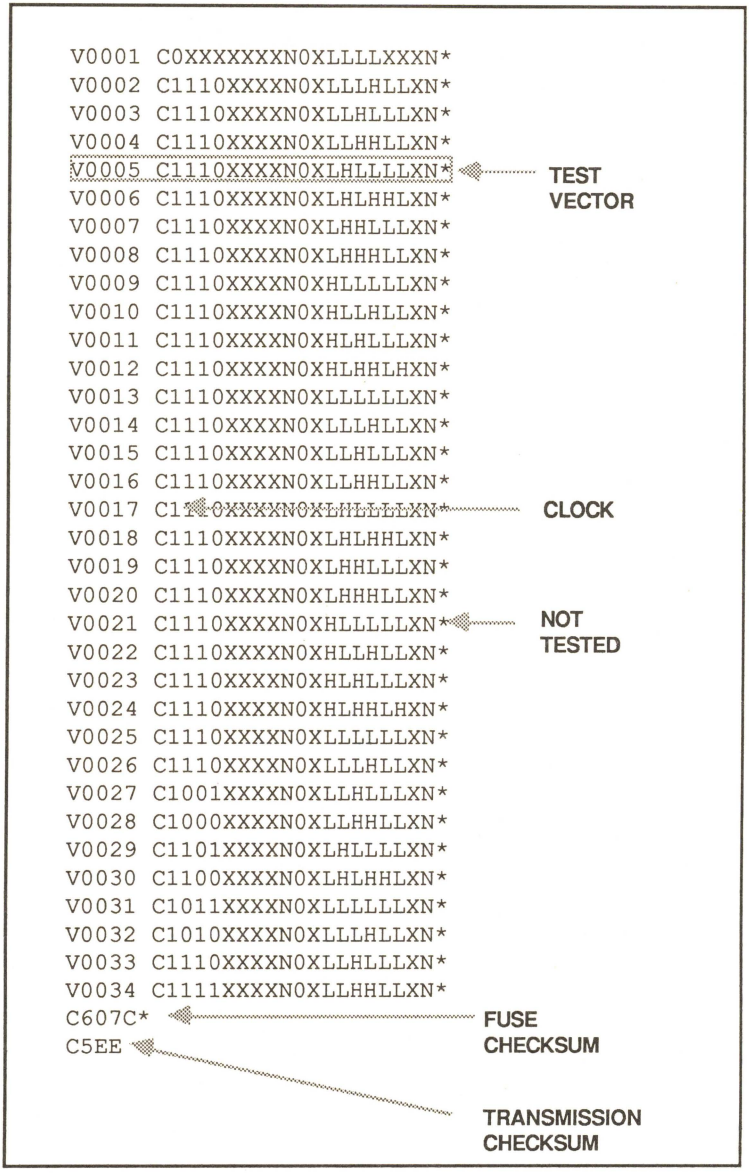
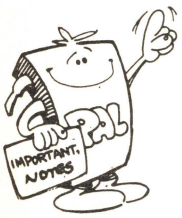



Figure 4-63

Test Vectors from SUPER.JDC


IMPORTANT
NOTES

# 7. Program the Device

### *About This Chapter*

This chapter outlines two methods to program a device:

| *To...* | *Read this Section...* |
|---|---|
| Use a computer to send JEDEC files to your programmer | 7.1 |
| Use PC2 communications software | 7.2 |
| Copy files from a programmed master device | 7.3 |
| Download the JEDEC file | 7.4 |

For detailed instructions, specific to your programmer, refer to your programmer manual.

7.1

## Send JEDEC Files To The Programmer

To program a device from a JEDEC file, complete the following tasks.

1. Set up the communications link between the programmer and your computer using either MS-DOS commands or the PC2 communications program.

2. Connect the programmer to your computer.

3. Send a JEDEC file to the programmer.

4. Program the device.

The sections that follow give detailed descriptions of each step.

7.1.1

## Connect the Programmer

To connect the programmer to the computer serial port, follow these steps.

1. Make sure the computer has a serial port. Most programmers require a serial connection to the computer.

2. Verify the device name of the serial port, COM1: or COM2: (You will need to know this when you establish the communications link).

3. Connect the programmer to a serial port on the computer. Use the cable specified in the programmer manual. Figure 4-64 shows a typical programmer-to-computer connection.

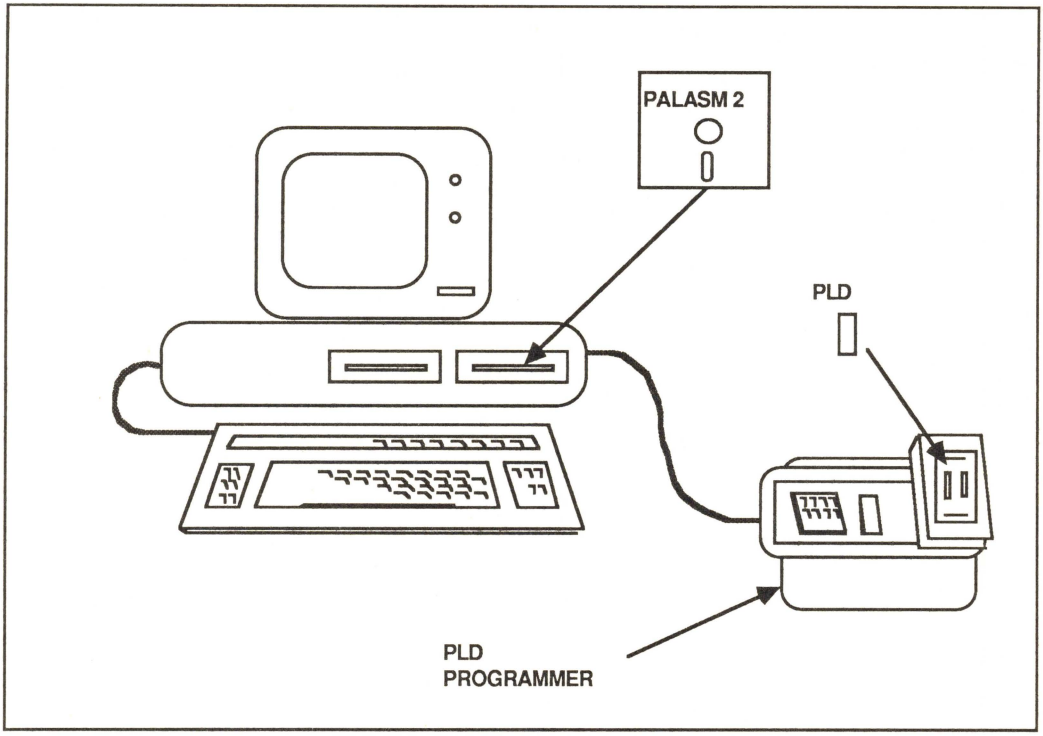



Figure 4-64

Connect the Programmer to a Computer

7.1.2

## Set Up The Communications Link

To set up the programmer transmission parameters, follow these steps.

1. Set the transmission parameters for the programmer. Refer to the programmer manual for instructions. If these parameters are fixed in your programmer, note the settings so that you can configure the computer to match.

2. Set the transmission parameters for the computer. You can use:

   - MS-DOS commands
   - PC2, a software communications program supplied on the supplemental disk
   - Any commercially available communications software

| *To use...* | *You will...* |
|---|---|
| MS-DOS commands | • Type MODE followed by the device name, baud rate, parity, number data bits, number stop bits. For example,<br><br>MODE COM1:4800,N,7,1<br><br>To ensure a reliable transfer, use 4800 baud or lower.<br><br>• Proceed to *Transmit The JEDEC File Using MS-DOS*, Section 7.1.3 |
| PC2 software program | Skip to *PC2 Communications Software,* Section 7.2 |
| Commercial communications software | Refer to that communications software manual for instructions. |

7.1.3

## Transmit The JEDEC File Using MS-DOS

The PALASM 2 software generates different JEDEC files, depending on which programs you run. If you run the XPLOT program, the software generates fuse maps in a file with the extension .JED. The programmer uses the .JED file to program the device.

If you run the SIM program after XPLOT, the software adds test vectors to the .JED file and generates a file with the extension .JDC. Many programmers can use the test vectors in the .JDC file to perform a functional test after programming and verification.

**Note:** On PROSE and PLS devices the PROASM and PLSASM respectively are the assembly programs that create the .JED file.

To prepare the programmer to receive a .JDC or JED file, follow these steps.

1. Set up the programmer to receive the JEDEC files. Refer to your programmer manual for instructions.

2. Use the MS-DOS command, COPY, to send the JEDEC file to the serial port:

   COPY filename COM1:

   The programmer should indicate that it is receiving the file.

3. Make sure the programmer successfully received the file. The programmer should indicate that the transmission is complete.

4. Proceed to *Download The JEDEC File*, Section 7.4.

7.2

## PC2 Communications Software

With PC2 software, a bidirectional communications program, you can set up and verify the communications link between the programmer and the IBM-PC/XT/AT. You can also set up computer transmission parameters and send a JEDEC file to the programmer.

7.2.1

## Load PC2

You can load the PC2 software program from a hard disk drive or from floppy disks. This section describes both methods.

7.2.1.1

### Load PC2 From A Hard Disk Drive

1. Make sure PC2 is installed on your hard disk drive.

| *If you...* | *Then...* |
|---|---|
| Already installed PC2 | Proceed to step 2. |
| Did not install PC2 | Follow the instructions in *Install The Menu,* Section 2.1.3.1. When the screen displays the input install request menu, select option 5, Install Supplementary Software only.<br><br>After you install the supplementary programs and exit the input install request menu, the screen displays the system prompt. |

2. From the system prompt, start the PALASM 2 software main menu by entering

   PALASM <return>

   and press <return> again.

3. Use the PALASM 2 software main menu to start PC2. From the main menu, press <F4> Program Device.

   F4 Program Device

   The screen displays the function key menu, as shown in Figure 4-65. For a description of the function keys, proceed to *PC2 Function Keys Defined*, Section 7.2.1.3.

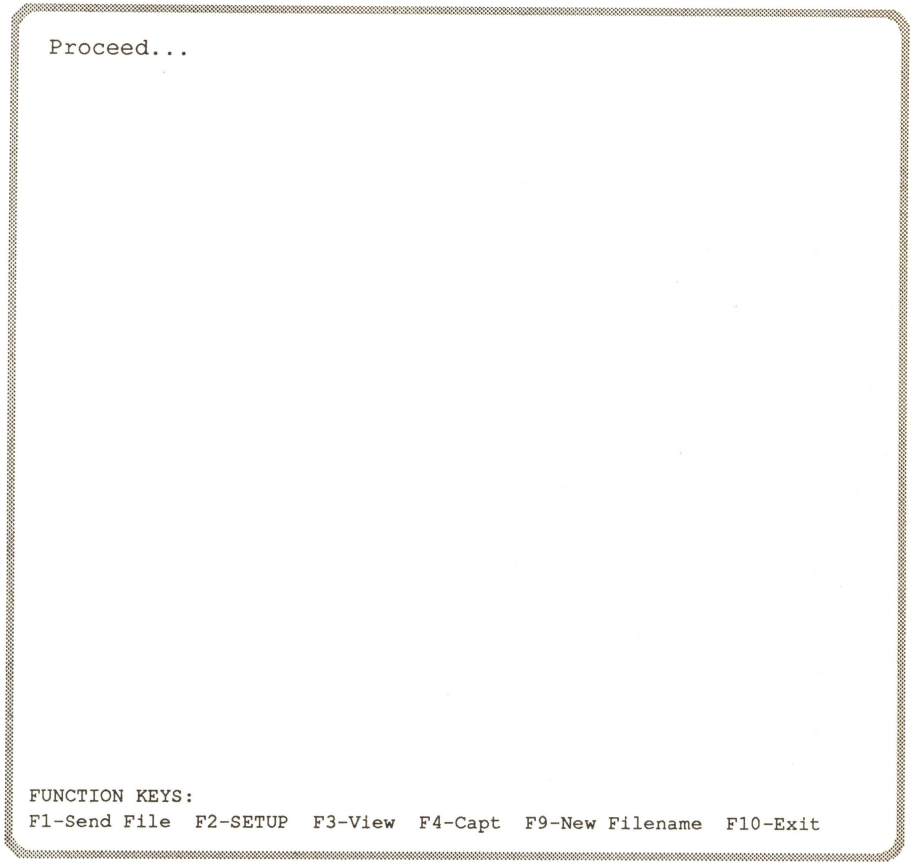



Figure 4-65

PC2 Function Key Menu Screen

7.2.1.2

## Load PC2 From A Two Floppy System

To load PC2 on a two floppy disk drive system, follow these steps.

1. Insert the backup copy of the Supplementary disk in drive A.
2. Insert the data disk containing your JEDEC file in drive B.

3. Type PC2 and press <return>. The screen displays the function key menu, as shown in Figure 4-65.

4. For a description of the function keys, proceed to *PC2 Function Keys Defined*, Section 7.2.1.3.

7.2.1.3

## PC2 Function Keys

Table 4-27 describes the use of each function key.

**Table 4-27**

**PC2 Function Keys**

| Function Key | Function |
|---|---|
| <F1> | Send a file |
| <F2> | Configure the communications port |
| <F3> | View the JEDEC file as it is sent to the programmer |
| <F4> | Save the transmitted data to a disk file |
| <F9> | Name a new file |
| <F10> | Exit the PC2 program |

7.2.2

## Set Up Computer Transmission Parameters

To set up transmission parameters for your computer, follow these steps.

1. To display the setup menu, press <F2>.

2. To view all the options for a selected parameter, press the spacebar. A circle (bullet) indicates the selected parameter.

3. When the correct option displays, press <return> to advance to the next parameter.

4. Repeat steps 2 and 3 for each parameter.

5. After you select an option for the last parameter (Stop bits) and press <return>, the screen displays the transmission parameters. Figure 4-66 shows a sample setup.

```
SETUP PROGRAM FOR PALASM COMMUNICATION...
    <CR> Next Field,  <SPACE> Modify

      o  Com port. . . . . : COM1:
         Baud rate . . . . . : 2400
         Parity . . . . . .: E
         Data bits . . . . . : 7
         Stop bits . . . . . : 1

New values ok (Y/N) ? _

(C)Copyright 1983,1984 Monolithic Memories
           All Rights Reserved

FUNCTION KEYS:
F1-Send File  F2-SETUP  F3-View  F4-Capt  F9-New Filename  F10-Exit
```

**Figure 4-66**

**Computer Transmission Parameters Screen**

7. Decide whether you want to use the new parameter values by selecting Y or N:

| *If you type...* | *Then...* |
|---|---|
| <N> <return> | You can change any parameter setting. Repeat steps 2 and 3. |
| <Y> <return> | The screen displays:<br>`Make these changes permanent (Y/N) ?`<br>Proceed to step 7. |

7. Decide whether you want to save the changes to disk by selecting Y or N:

| *If you type...* | *Then...* |
|---|---|
| <N> <return> | You can change any parameter. Repeat steps 1 through 6. |
| <Y> <return> | The program saves these changes to the file, PC2.DAT, on the Supplementary diskette in drive A. The function key menu displays across the bottom of the screen (as in Figure 4-65). |

7.2.3

## Transmit The JEDEC File

To send a JEDEC file to your programmer, follow these steps.

1. To transmit one file, or the first of a series of files, to your programmer, press <F1>. Figure 4-67 shows the screen display.

2. To select another file to transmit to the programmer, press <F9>. Figure 4-67 shows the screen display.

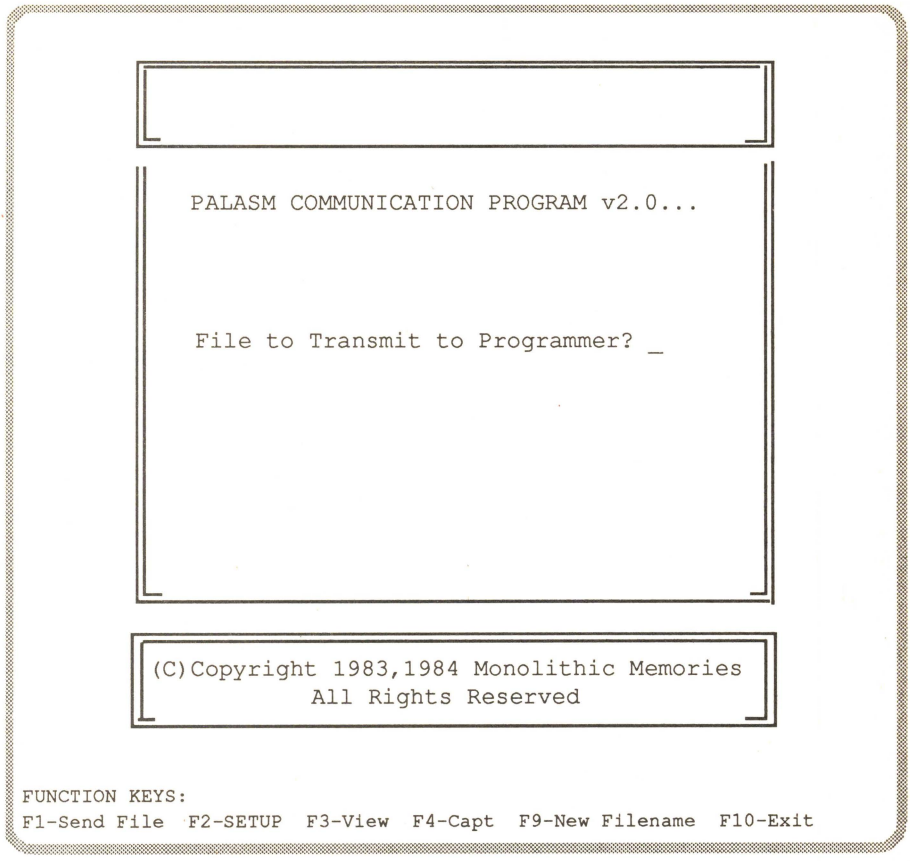



**Figure 4-67**

**PC2 Name Transmit File Screen**

2. Enter the drive name followed by the name of your JEDEC file and press <return>. The program verifies that the filename exists. When complete, the menu bar displays across the bottom of the screen.

3. Set up the programmer to receive the JEDEC file. Refer to your programmer manual for instructions.

4. To send the JEDEC file, press <F1>. Watch for this activity:

   - The screen displays: `Transmitting...`
   - The programmer indicates that it is receiving the file.
   - When complete, the screen displays: `End Transmission...`

5. If the transmission is not successful, check that the transmission parameters for the computer and the programmer match. Also, refer to your programmer manual for more troubleshooting information.

6. To exit the PC2 program, press <F10>. Figure 4-68 shows the screen display.

7. Proceed to *Download The JEDEC File*, Section 7.4.

7.3

## Copy From A Master Device

If you have a master device and you want to program a device of the same type with exactly the same pattern, follow these steps.

1. Set the programmer to read (or copy) the master device. You may also need to know the product code and device type specific to the manufacturer.

2. Install the correct adapter, if required.

3. Enter the appropriate product code or select the product from the menu. Refer to the programmer manual for instructions.

4. Install the programmed master device in the correct socket and read its fuse pattern into the programmer memory.

   The pattern is now in the programmer memory and will remain there unless the memory is cleared or the programmer power is turned off.

5. Verify that the checksum displayed at the end of the copy operation matches the checksum of the master device.

6. If the checksums match, you are now ready to program the device. Proceed to *Download The JEDEC File*, Section 7.4.

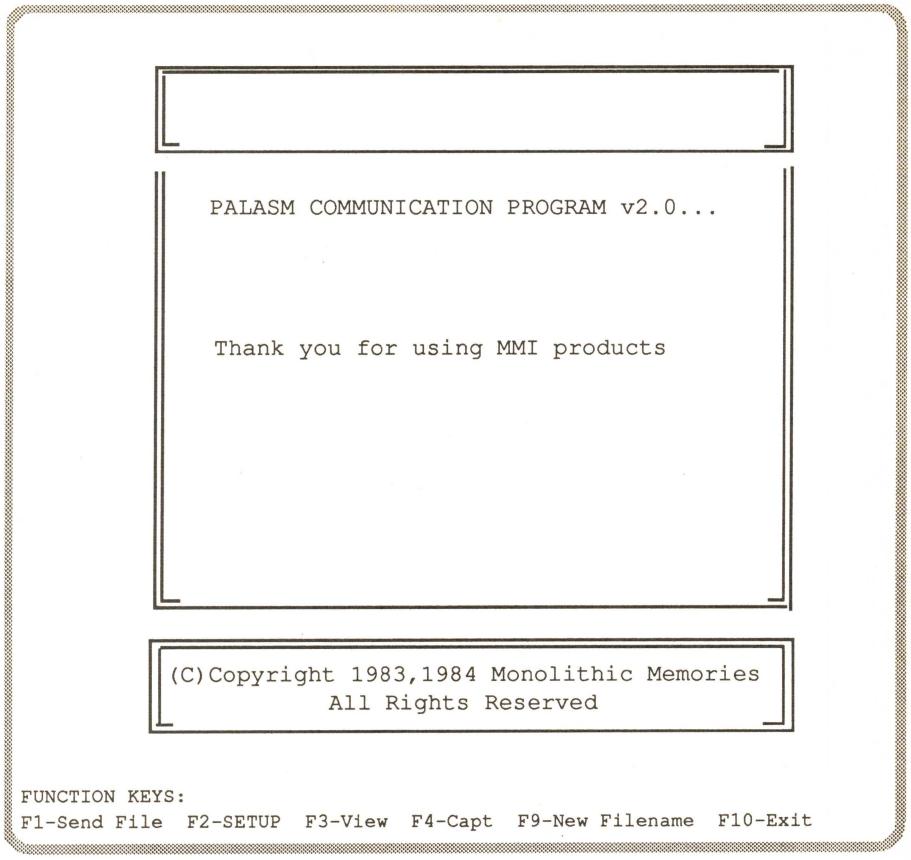



**Figure 4-68**

**PC2 Exit Screen**

7.4

## Program The Device

To program the device, follow these steps.

1. Follow the manufacturer's instructions to install a device into the programmer. For some programmers, you must know the device code and the correct adapter or software version.

2. Program the device by sending the file (now resident in the programmer) to the device.

With most programmers, you can specify the programmer to verify that the device was programmed correctly. If you run the SIM program, the software includes JEDEC test vectors in your file. The programmer uses these test vectors to perform a functional test.

# PALASM 2 Software Glossary

**Autorun:** A feature of the PALASM main menu that allows you to run the assembly and simulation programs with one keystroke.

**BINHEX supplementary program:** A binary to hexadecimal conversion program.

**Boolean equation design:** Specifies Boolean logic functions for programming a device to perform specified tasks and give specified outputs.

**CHECK command:** Compares the expected values and the simulated values of signals during simulation.

**CLOCKF command:** Generates a clock signal on the dedicated clock pin(s).

**Combinatorial equations:** Component of the Equations segment. Equations that combine signals for immediate output.

**Commands:** See individual command name.

**Condition:** A set of inputs in a state diagram.

**Conditions segment:** A segment of the state machine design that defines the input values that determine state transitions. The condition equations assign names to unique sets of inputs.

**Declaration segment:** Describes design identification, device and pin data, and string substitutions.

**DECODE supplementary program:** Address decoder program that generates PALASM 2 software Boolean equations.

**Editor:** Also called a word processor. A computer program that permits selective revision of computer-stored data.

**Equations segment:** A segment of the input file. Contains equations for Boolean and programmable functions that define outputs in terms of inputs and feedback.

**EXPAND program:** Expands input equations and converts state machine syntax to Boolean equations.

**Fields:** Areas on the PALASM 2 main menu where you enter data--specifically, the input PDS file name and the directory where that file is located.

**Files:** See individual filename.

**FOR...TO...DO loop command:** Optional construct in the Simulation segment syntax. Iterates a set of commands a fixed number of times.

**Functional equation:** Components of the Equations segment of a Boolean design. Defines these programmable functions: clock, set, reset, three-state, dynamic registered/combinatorial output selection.

**Fuse map:** An output file generated by the program XPLOT. Displays the programmed and unprogrammed fuses specified by the input file.

**Fuseplot:** See fuse map.

**History file:** An output file generated by the SIM program that shows the values of every pin through a simulation sequence.

**.HST file:** Simulation history data file.

**IF...THEN...ELSE command:** Conditional branching construct for simulation.

**Intermediate files:** Files created by the software but not immediately visible to the user.

**.JDC file:** JEDEC fuse data and JEDEC test vectors file.

**.JED file:** JEDEC fuse data file.

**JEDEC file:** Created by XPLOT and used by the device programmer to program a device.

**JEDMAN program:** Disassembles JEDEC files and generates Boolean equation input files. JEDMAN allows you to read a fuseplot directly from a programmed device.

**Keyword:** A word used by the software to identify the block of information that follows it.

**Latched equation:** Component of the Equations segment of a Boolean design. Defines logic functions for devices with latched outputs.

**Mealy state machine:** Determines its outputs from the inputs and the present state.

**MINIMIZE program:** Uses the intermediate file created by PARSE or EXPAND to perform automatic logic reduction.

**Moore state machine:** Determines its outputs from the present state only.

**Output equation:** Defines the state machine's operation in terms of conditions and outputs from the present state. The syntax for output equations is different for Mealy and Moore machines.

**PALASM2:** PALASM 2 software interactive menu program that simplifies user interface to the software.

**PARSE program:** Checks the syntax of the input file.

**PC2 supplementary program:** A menu-driven, multiple choice program that enables communication between PLD programmers and IBM-PC/XT/AT computers.

**.PDF file:** PLD architecture description data file.

**.PDS file:** User-defined PLD design input file.

**PDSCNVT supplementary program:** Converts previous PALASM version input files to PALASM 2 software syntax.

**PINOUT supplementary program:** Generates a list of the pin names from the .TRE file (created by the PALASM 2 software assembler).

**.PL2 file:** PDS file reconstructed from JEDEC output file.

**PLSASM program:** Assembles PLS device designs.

**PRLDF command:** Assigns logical values to, or initializes, register outputs in Simulation for preloadable PAL devices.

**PROASM-PROSIM:** Assembles and simulates PROSE device designs. PROASM accepts only state machine designs and generates a fusemap and a JEDEC file. PROSIM generates history and trace files as well as JEDEC test data.

**Programmed fuse:** Equivalent to a "1" in a JEDEC file. Sometimes refered to as a "blown." fuse. See also Unprogrammed fuse.

**Programs:** See individual program name.

**Registered equation:** Component of the equations segment of a Boolean design. Defines logic functions for devices with registered outputs.

**Reserved word:** A word used by the software to identify design segments and information, device codes, commands, functions, and pin defaults. Some reserved words are keywords that identify the block of information that follows.

**Run-time log:** Contains the messages or intermediate files generated after running each program in the early stages of processing.

**SCRSIM supplementary program:** Generates simulation waveforms from history and trace output files. These waveforms can be viewed on the screen or sent to a printer.

**SETF command:** Specifies new input values for the software to simulate. Also used to control the three-state function in simulation.

**SIM program:** Checks the functionality of a PLD device. SIM simulates the operation of your design, calculating the output values based on input signals defined in the Simulation segment. After running PARSE, EXPAND, and MINIMIZE, SIM generates two output files: a history file and a trace file.

**Simulation segment:** A segment of the design. Defines a trial set of inputs for a design and tells the software what to do with them.

**State assignment:** An equation that defines a state as a unique combination of outputs. Also called "state bit assignment" and "bit assignment."

**State diagram:** Illustrates the behavior of a state machine. Includes: all named states, the input values that cause state transitions when a clock pulse occurs, and the output values expected because of state transitions.

**State equation:** Defines the states in terms of conditions that determine transitions to other states. State equations are necessary for both Mealy and Moore designs.

**State segment:** A segment of the state machine design. Contains information about the design and equations that describe how the machine functions. Describes defaults, pin assignments to states, and equations for state transitions and outputs.

**State machine design:** An input file that contains information for programming a device to cycle through defined states and give specified outputs.

**Supplementary programs:** Programs included with the PALASM 2 software package but not supported by Monolithic Memories. See also individual supplementary program name.

**TIMING supplementary program:** Timing diagram entry program.

**Trace file:** A subset of the history file that shows only the pins you specify between TRACE_ON and TRACE_OFF simulation commands. See also History file.

**TRACE_OFF command:** Turns off the TRACE_ON command. After this command, no more results are added to the trace file until the next TRACE_ON command appears.

**TRACE_ON command:** Defines specific signals whose values will be recorded in the simulation trace file.

**TREPL2 program:** Disassembles intermediate files created by PARSE, EXPAND, and MINIMIZE, and converts them to Boolean equation input files.

**.TRF file:** Simulation trace data file.

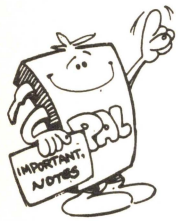
**Unprogrammed fuse:** Equivalent to a "0" in a JEDEC file. Sometimes refered to as "intact." See also Programmed fuse.

**VTRACE supplementary program:** A utility program to convert simulation output files to timing diagrams.

**WHILE...DO loop command:** Optional construct of the simulation syntax that iterates a set of commands until a condition is satisfied.

**XPLOT program:** After PARSE, EXPAND, and MINIMIZE, assembles PAL device designs. Validates the architectural design of an input PLD design containing Boolean equations (created by EXPAND or MINIMIZE) and produces fusemaps and JEDEC data.

**.XPT file:** PLD fuse map data file.


IMPORTANT
NOTES

# PALASM 2 Software Index

## A, B


Assemble
- autorun feature **4-36**
- input file, procedure **4-30, 38**

Autorun assembly
- how to use **4-36**

BINHEX **4-12, 26**

Boolean equations
- convert from state machine syntax **4-40, 41**
- declaration segment **4-65**
- design file structure **4-62**
- equations **4-70**
- how to expand **4-40**
- how to minimize **4-41**
- syntax rules **4-63, 138**


## C


CHECK command
- defined **4-140**
- syntax **4-146**
- when to use **4-145**

CHIP
- information for state and output equations **4-118**
- keyword defined **4-109**
- syntax **4-67, 109**

CLOCKF command
- clocking procedure **4-156**
- defined **4-140**
- syntax **4-144**
- the c character **4-156**

Combinatorial equations
- polarity **4-71**
- syntax **4-70**

Computers supported **4-4**

Conditions segment
- keyword **4-125**
- syntax **4-125**
- when conditions conflict **4-126**

## D, E


Data entry fields
- defined **4-32**
- directory **4-35**
- input PDS file **4-35**

Declaration segment
- Boolean **4-65**
- CHIP syntax **4-67, 109**
- design header **4-66, 108**
- in a Boolean equation design **4-62**
- in a state machine design **4-104**
- keywords **4-66, 108**
- STRING syntax **4-68, 110**

DECODE **4-12, 26**

DeMorgan's theorem
- when to use **4-125**

Devices supported **4-2**

DOS
- commands to transmit JEDEC file **4-173**
- how to enter from menu **4-35**
- to run PALASM 2 software **4-59**

EQUATIONS **4-70**

Equations segment
- combinatorial equations syntax **4-70**
- functional equations
  - programmable set and reset **4-74**
  - programmable three-state **4-75**
- functional equations **4-74**
- in a Boolean equation **4-62, 70**
- `keywords **4-70**
- registered equations syntax **4-72**

Error detection
- view run-time log **4-53**

EXPAND
- input equations, procedure **4-30, 40**
- program for PAL devices **4-130**

## F

FOR...TO...DO loop
- defined **4-140**
- syntax **4-148**

Function keys
- defined **4-32**
- display directory (F1) **4-35**
- edit PDS input file (F3) **4-36**
- enter DOS (F2) **4-35**
- for PC2 **4-176**
- PALASM2 option (F5) **4-37, 38, 46**
- program device (F4) **4-174**
- view data (F7) **4-44, 47, 50**

Functional equations
- defined **4-74**
- global set and reset syntax **4-75**
- programmable set and reset syntax **4-75**
- programmable three-state syntax **4-75**
- syntax for PAL16RA8 and PAL20RA10 **4-85**

Fuse map
- how to interpret **4-56**
- stored in .XPT file **4-42**
- view .XPT output file **4-43**

## H, I

History file
- .HST filename **4-140, 147**
- for a PROSE device **4-163**
- simulation segment output file **4-140**
- TRACE_OFF command to create time frames **4-147**

History waveform
- defined **4-157**
- how to view **4-48**

IF...THEN...ELSE command
- defined **4-140**
- syntax **4-149**

Input file
- assemble **4-38, 42**
- check syntax **4-38**
- expand equations **4-40**
- minimize equations **4-41**

Install PALASM 2 Software **4-15**

## J, K

JED file
- created by assembler **4-56**
- fuse maps in **4-173**
- test vectors in **4-173**
- output file generated **4-42**
- view JEDEC fuse data **4-43**

JEDEC file
- convert to Boolean equation input file **4-50**
- disassembly program **4-50**
- how to generate as output **4-30**
- how to interpret **4-56**
- how to interpret test data **4-166**
- program a device from **4-170**
- required to program device, how to generate **4-42**
- test data, how to view **4-50**
- view fuse data **4-43**

JEDMAN
- defined **4-10**
- program to disassemble JEDEC file **4-52**

Keywords
- as reserved words in a boolean design **4-63**
- as reserved words in a state machine design **4-104**
- CHIP **4-108**
- CONDITIONS **4-125**
- EQUATIONS **4-70**
- global default options **4-115**
- in sample input file **4-36**
- SIMULATION **4-139**
- STATE **4-112**
- STRING **4-68, 110**

## M

Mealy state machine
- combinatorial output equation syntax **4-121**
- conditions defined **4-97**
- functional state diagram **4-97**
- global default options **4-115**
- how output is determined **4-96**
- registered output equation syntax **4-120**
- simplify a state diagram **4-99**

## M (Continued)

Menu
- data entry fields defined **4-32**
- function keys defined **4-32**
- PALASM main, how to start **4-32**
- status line defined **4-33**

Minimize
- input equations, procedure **4-30, 41**

MINIMIZE
- program for PAL devices **4-130**

Moore state machine
- combinatorial output syntax **4-121**
- conditions defined **4-101**
- functional state diagram **4-100**
- how output is determined **4-100**
- registered output syntax **4-121**
- simplify a state diagram **4-102**

## O, P

Output equations
- combinatorial Mealy output syntax **4-121**
- defined **4-120**
- in a state segment **4-117**
- registered Mealy output syntax **4-120**

PAL device
- .CMBF syntax **4-91**
- buried registers **4-87**
- designs with XOR gates **4-83**
- event-driven simulator **4-137**
- global preset syntax **4-84**
- global reset syntax **4-85**
- internal XOR gates **4-87**
- product terms **4-83, 88**
- programmable polarity, **4-71**
- state machine design considerations **4-130**
- syntax for PAL16RA8 and PAL20RA10 functional equations **4-85**

## P (Continued)

PALASM 2 Software
- add supplementary programs **4-25**
- event-driven simulator **4-137**
- input files **4-11**
- install **4-15**
- intermediate files **4-11**
- output files **4-11**
- programs
  - JEDMAN **4-10**
  - PALASM **4-7**
  - PARSE **4-8**
  - PLSASM **4-10**
  - PROASM-PROSIM **4-9**
  - SIM **4-9**
  - XPLOT **4-9**
- reserved words in a Boolean design **4-64**
- reserved words in a state machine design **4-106**
- run the software
  - from DOS **4-59**
  - procedure to **4-30**
- setup **4-23**
- supported computers **4-4**
- supported devices **4-2**
- TRE files **4-54**

PALASM2 menu option
- assemble input file **4-42**
- autorun assembly **4-36**
- check design file syntax **4-38**
- disassemble TRE file **4-55**
- expand input equations **4- 40**
- minimize input equations **4- 41**

PC2
- how to exit **4-180**
- how to load **4-174**
- supplementary program **4-12**
- use to set transmission parameters **4-172**
- using **4-173**

PINOUT **4-12, 26**

PLS device
- complement array **4-81**
- output equation syntax **4-81**
- pin preset syntax **4-82**
- state machine design considerations **4-128**

## P (Contined)

Polarity
- factors determining output **4-76**
- how to determine output **4-78**
- in combinatorial equations **4-71**
- output defined **4-76**
- programmable **4-71**
- summary for active-low outputs **4-78**

PRLDF command
- defined **4-140**
- guidelines **4-141**
- syntax **4-141**
- the p character **4-157**

PROASM-PROSIM **4-9**
Program the device *(See PC2)*
PROSE device
- history file **4-163**
- state machine design considerations **4-129**

## R

Registered equations
- syntax **4-72**

Reserved words
- equations in a Boolean design **4-62**
- global default options **4-115**
- in a state machine design **4-104**
- list, for a Boolean design **4-64**
- list, for a state machine design **4-106**

RSTF command **4-74**
Run-time log
- print **4-53**
- view for error detection **4-53**
- view TRE file **4-55**

## S

SCRSIM **4-13, 26**
SETF command
  defined **4-140**
  syntax **4-142**
  the g character **4-156**
SIM **4-9**
Simulate
  output files **4-140**
  procedure **4-46**
  view output files **4-47**
Simulation
  commands
    CHECK **4-140**
    FOR...TO...DO loop **4-140**
    IF...THEN...ELSE**4-140**
    PRLDF **4-140**
    SETF **4-140**
    TRACE_OFF**4-140**
    TRACE_ON **4-140**
    WHILE...DO loop **4-140**
  constructs
    FOR loop **4-148**
    IF...THEN...ELSE loop **4-149**
    WIHLE...DO loop **4-148**
  syntax
    CHECK **4-146**
    CLOCKF **4-144**
    FOR...DO loop **4-148**
    IF...THEN...ELSE loop **4-149**
    PRLDF **4-141**
    SETF **4-142**
    TRACE_ON **4-147**
    WHILE  DO loop **4-148**

## S (Continued)


Simulation segment
- CHECK command defined **4-145**
- CLOCKF syntax **4-144**
- commands **4-140**
- defined **4-138**
- FOR TO DO loop defined **4-148**
- IF THEN ELSE command defined **4-149**
- keyword **4-139**
- language directives **4-139**
- output files **4-140**
- PRLDF command syntax **4-141**
- SETF syntax **4-142**
- TRACE_OFF command defined **4-147**
- TRACE_ON command defined **4-147**
- WHILE DO loop defined **4-148**

State assignment defined **4-116**

State machine
- design structure **4-103**
- design syntax rules **4-105, 138**
- diagram requirements to build a design **4-96**
- equations
  - in a state segment **4-117**
  - transition syntax **4-118**
- Mealy output **4-96**
- Moore output **4-100**
- state and output equations **4-117**
- syntax to convert to Boolean equations **4-41**

State segment
- in a state machine design **4-112**
- keywords **4-112**
- state assignments syntax **4-116**

STRING
- keyword defined in a Boolean design **4-68**
- keyword defined in a state machine design **4-110**
- syntax **4-68, 110**

SUPER.PDS
- example input file demonstration **4-29**
- how to open **4-36**
- verify file location **4-35**

## S (Continued)

Supplementary Programs
- BINHEX **4-12, 26**
- DECODE **4-12, 26**
- PC2 **4-12**
- PDSCNVT **4-12**
- PINOUT **4-12, 26**
- SCRSIM **4-13, 26**
- TIMING **4-12, 26**
- VTRACE **4-12, 26**

## T

TIMING **4-12, 26**
Trace file
- .TRF filename **4-140, 147**
- how different from history file **4-165**
- simulation segment output file **4-140**
- TRACE_ON command to define signal values **4-147**

Trace waveform
- defined **4-159**
- how different from history waveforms **4-161**
- how to view **4-48**
- TRACE_OFF command **4-140, 147**

TRACE_ON command
- defined **4-140**
- syntax 4- **147**
- to record trace waveforms **4-159**

TRE file
- convert to a Boolean equation **4-55**
- how to disassemble **4-55**
- view run-time log **4-55**
- when PALASM 2 software creates **4-54**

## V, W, X

VTRACE **4-12, 26**
WHILE DO loop
- defined **4-140**
- syntax **4-148**

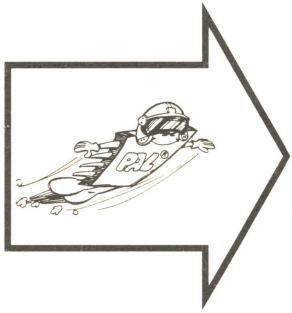
XPLOT **4-8**
XPT file
- created by assembler **4-56**
- to store fuseplot **4-42**

## PAL Device Handbook


Introduction 1

Applications 2


## PAL Device Data Book




Programming and Quality 3

PALASM 2 Software User Documentation 4

Data Sheets 5

Appendices 6

**PAL/PLD Device Menu** ..... 5-3

**TTL/CMOS PAL Devices** ..... 5-9

PAL16RA8 ..... 5-11

PAL16RP8A Series ..... 5-17
- PAL16P8A
- PAL16RP8A
- PAL16RP6A
- PAL16RP4A

PAL16R8 Family ..... 5-26

PAL16R8D Series ..... 5-29
- PAL16L8D
- PAL16R8D
- PAL16R6D
- PAL16R4D

PAL16R8B Series ..... 5-31
- PAL16L8B
- PAL16R8B
- PAL16R6B
- PAL16R4B

PALC16R8Q-25 Series ..... 5-33
- PAL16L8Q-25
- PAL16R8Q-25
- PAL16R6Q-25
- PAL16R4Q-25

PAL16R8B-2 Series ..... 5-35
- PAL16L8B-2
- PAL16R8B-2
- PAL16R6B-2
- PAL16R4B-2

PAL16R8A Series ..... 5-37
- PAL16L8A
- PAL16R8A
- PAL16R6A
- PAL16R4A

PAL16R8B-4 Series ..... 5-39
- PAL16L8B-4
- PAL16R8B-4
- PAL16R6B-4
- PAL16R4B-4

PAL16R8A-2 Series ..... 5-41
- PAL16L8A-2
- PAL16R8A-2
- PAL16R6A-2
- PAL16R4A-2

PAL16R8A-4 Series ..... 5-43
- PAL16L8A-4
- PAL16R8A-4
- PAL16R6A-4
- PAL16R4A-4

PALC16R8Z-25 Series ..... 5-50
- PAL16L8Z-25
- PAL16R8Z-25
- PAL16R6Z-25
- PAL16R4Z-25

PAL16X4 ..... 5-51

PAL10H8 Series ..... 5-56
- PAL10H8
- PAL12H6
- PAL14H4
- PAL16H2
- PAL16C1
- PAL10L8
- PAL12L6
- PAL14L4
- PAL16L2

PAL32VX10A ..... 5-70

PAL32VX10 ..... 5-70

PALC22V10H-25 ..... 5-79

PALC22V10H-35 ..... 5-79

PAL22RX8A ..... 5-87

PAL20RA10-20 ..... 5-95

PAL20RA10 ..... 5-97

PAL20RS10 Series ..... 5-103
- PAL20S10
- PAL20RS10
- PAL20RS8
- PAL20RS4

PAL20X10A Series ..... 5-113
- PAL20L10A
- PAL20X10A
- PAL20X8A
- PAL20X4A

PAL20R8 Family ..... 5-122
PAL20R8B Series ..... 5-125
PAL20L8B
PAL20R8B
PAL20R6B
PAL20R4B
PAL20R8B-2 Series ..... 5-126
PAL20L8B-2
PAL20R8B-2
PAL20R6B-2
PAL20R4B-2
PAL20R8A Series ..... 5-128
PAL20L8A
PAL20R8A
PAL20R6A
PAL20R4A
PAL20R8A-2 Series ..... 5-130
PAL20L8A-2
PAL20R8A-2
PAL20R6A-2
PAL20R4A-2
PALC20R8Z-35 Series ..... 5-133
PALC20L8Z-35
PALC20R8Z-35
PALC20R6Z-35
PALC20R4Z-35
PALC20R8Z-45 Series ..... 5-133
PALC20L8Z-45
PALC20R8Z-45
PALC20R6Z-45
PALC20R4Z-45

PAL6L16A ..... 5-141
PAL8L14A ..... 5-141

PAL12L10 Series ..... 5-147
PAL12L10
PAL14L8
PAL16L6
PAL18L4
PAL20L2
PAL20C1

PAL32R16 ..... 5-158

General Information ..... 5-164

**TTL/CMOS AmPAL Devices** ..... 5-167

AmPAL23S8-20 ..... 5-169
AmPAL23S8-25 ..... 5-169

AmPAL16R8 Family ..... 5-184
AmPAL16R8D Series ..... 5-183
AmPAL16L8D
AmPAL16R8D
AmPAL16R6D
AmPAL16R4D
AmPAL16R8B Series ..... 5-197
AmPAL16L8B
AmPAL16R8B
AmPAL16R6B
AmPAL16R4B
AmPAL16R8AL Series ..... 5-197
AmPAL16L8AL
AmPAL16R8AL
AmPAL16R6AL
AmPAL16R4AL
AmPAL16R8A Series ..... 5-197
AmPAL16L8A
AmPAL16R8A
AmPAL16R6A
AmPAL16R4A
AmPAL16R8Q Series ..... 5-197
AmPAL16L8Q
AmPAL16R8Q
AmPAL16R6Q
AmPAL16R4Q
AmPAL16R8L Series ..... 5-197
AmPAL16L8L
AmPAL16R8L
AmPAL16R6L
AmPAL16R4L
AmPAL16R8 Series ..... 5-197
AmPAL16L8
AmPAL16R8
AmPAL16R6
AmPAL16R4

AmPAL18P8B ..... 5-202
AmPAL18P8AL ..... 5-202
AmPAL18P8A ..... 5-202
AmPAL18P8Q ..... 5-202
AmPAL18P8L ..... 5-202

5

AmPALC29MA16-35 ..... 5-209
AmPALC29MA16-45 ..... 5-209
AmPALC29M16-35 ..... 5-231
AmPALC29M16-45 ..... 5-231

AmPAL22V10-15 ..... 5-249
AmPAL22V10A ..... 5-260
AmPAL22V10 ..... 5-260

AmPAL20XRP10 Family ..... 5-271
AmPAL20XRP10-20 Series ..... 5-286
AmPAL22XP10-20
AmPAL20XRP10-20
AmPAL20XRP8-20
AmPAL20XRP6-20
AmPAL20XRP4-20
AmPAL20XRP10-30L Series ..... 5-286
AmPAL22XP10-30L
AmPAL20XRP10-30L
AmPAL20XRP8-30L
AmPAL20XRP6-30L
AmPAL20XRP4-30L
AmPAL20XRP10-30 Series ..... 5-286
AmPAL22XP10-30
AmPAL20XRP10-30
AmPAL20XRP8-30
AmPAL20XRP6-30
AmPAL20XRP4-30
AmPAL20XRP10-40L Series ..... 5-286
AmPAL22XP10-40L
AmPAL20XRP10-40L
AmPAL20XRP8-40L
AmPAL20XRP6-40L
AmPAL20XRP4-40L

AmPAL20RP10 Family ..... 5-291
AmPAL20RP10B Series ..... 5-306
AmPAL22P10B
AmPAL20RP10B
AmPAL20RP8B
AmPAL20RP6B
AmPAL20RP4B
AmPAL20RP10AL Series ..... 5-306
AmPAL22P10AL
AmPAL20RP10AL
AmPAL20RP8AL
AmPAL20RP6AL
AmPAL20RP4AL
AmPAL20RP10A Series ..... 5-306
AmPAL22P10A
AmPAL20RP10A
AmPAL20RP8A
AmPAL20RP6A
AmPAL20RP4A

AmPAL20L10B ..... 5-306
AmPAL20L10-20 ..... 5-306
AmPAL20L10AL ..... 5-306

**PROSE/PLS Sequencers** ..... 5-313
PMS14R21A ..... 5-315
PMS14R21 ..... 5-315

PLS167-33 ..... 5-331
PLS168-33 ..... 5-331
PLS105-37 ..... 5-331

**FPC/PEG Sequencers** ..... 5-337
Am29PL141 Fuse Programmable Controller ..... 5-339
Am2971 Programmable Event Generator ..... 5-365

**ECL PAL Devices** ..... 5-379
PAL10020EV/EG8 ..... 5-381
PAL10H20EV/EG8 ..... 5-381
PAL10H20G8 ..... 5-382
PAL10H20P8 ..... 5-385

**HAL/ZHAL Devices** ..... 5-391
ZHAL20A Series ..... 5-394
ZHAL24A Series ..... 5-401

**Military PAL Devices** ..... 5-415
Introduction ..... 5-417
Military PAL/PLD Device Menu ..... 5-418
Military 20-pin PAL Devices ..... 5-421
Military 24-pin PAL Devices ..... 5-439
DC/AC Parametric Testing ..... 5-469
JAN 38510 and Standard Military Drawings ..... 5-470
Military Screening ..... 5-474
Quality Programs ..... 5-477

**Logic Cell Array** ..... 5-481
M2064 ..... 5-483
M2018 ..... 5-483
Military M2064/M2018 ..... 5-518

**Electrical Definitions** ..... 5-531

# Alphanumeric Product Index


Am29PL141 5-339
Am2971 5-365
AmPAL16L8 5-197
AmPAL16L8A 5-197
AmPAL16L8AL 5-197
AmPAL16L8B 5-197
AmPAL16L8D 5-183
AmPAL16L8L 5-197
AmPAL16L8Q 5-197
AmPAL16R4 5-197
AmPAL16R4A 5-197
AmPAL16R4AL 5-197
AmPAL16R4B 5-197
AmPAL16R4D 5-183
AmPAL16R4L 5-197
AmPAL16R4Q 5-197
AmPAL16R6 5-197
AmPAL16R6A 5-197
AmPAL16R6AL 5-197
AmPAL16R6B 5-197
AmPAL16R6D 5-183
AmPAL16R6L 5-197
AmPAL16R6Q 5-197
AmPAL16R8 5-197
AmPAL16R8A 5-197
AmPAL16R8AL 5-197
AmPAL16R8B 5-197
AmPAL16R8D 5-183
AmPAL16R8L 5-197
AmPAL16R8Q 5-197

AmPAL18P8A 5-202
AmPAL18P8AL 5-202
AmPAL18P8B 5-202
AmPAL18P8L 5-202
AmPAL18P8Q 5-202

AmPAL20L10AL 5-306
AmPAL20L10B 5-306
AmPAL20L10-20 5-306
AmPAL20RP4A 5-306
AmPAL20RP4AL 5-306
AmPAL20RP4B 5-306
AmPAL20RP6A 5-306
AmPAL20RP6AL 5-306
AmPAL20RP6B 5-306
AmPAL20RP8A 5-306
AmPAL20RP8AL 5-306
AmPAL20RP8B 5-306
AmPAL20RP10A 5-306
AmPAL20RP10AL 5-306
AmPAL20RP10B 5-306
AmPAL20XRP4-20 5-286
AmPAL20XRP4-30L 5-286
AmPAL20XRP4-30 5-286
AmPAL20XRP4-40L 5-286
AmPAL20XRP6-20 5-286
AmPAL20XRP6-30L 5-286
AmPAL20XRP6-30 5-286
AmPAL20XRP6-40L 5-286
AmPAL20XRP8-20 5-286
AmPAL20XRP8-30L 5-286
AmPAL20XRP8-30 5-286
AmPAL20XRP8-40L 5-286
AmPAL20XRP10-20 5-286
AmPAL20XRP10-30L 5-286
AmPAL20XRP10-30 5-286
AmPAL20XRP10-40L 5-286

AmPAL22P10A 5-306
AmPAL22P10AL 5-306
AmPAL22P10B 5-306
AmPAL22V10 5-260
AmPAL22V10-15 5-249
AmPAL22V10A 5-260
AmPAL22XP10-20 5-286
AmPAL22XP10-30L 5-286
AmPAL22XP10-30 5-286
AmPAL22XP10-40L 5-286

AmPAL23S8-20 5-169
AmPAL23S8-25 5-169

AmPALC29M16-35 5-231
AmPALC29M16-45 5-231
AmPALC29MA16-35 5-209
AmPALC29MA16-45 5-209

M2018 5-483
M2064 5-483

PAL6L16A 5-141
PAL8L14A 5-141

PAL10H8 5-56
PAL10H20G8 5-382
PAL10H20EV/EG8 5-381
PAL10H20P8 5-385
PAL10L8 5-56

PAL12H6 5-56
PAL12L6 5-56
PAL12L10 5-147

PAL14H4 5-56
PAL14L4 5-56
PAL14L8 5-147

PAL16C1 5-56
PAL16H2 5-56
PAL16L2 5-56
PAL16L6 5-147
PAL16L8D 5-29
PAL16L8A 5-37
PAL16L8A-2 5-41
PAL16L8A-4 5-43
PAL16L8B 5-31
PAL16L8B-2 5-35
PAL16L8B-4 5-39
PAL16P8A 5-17
PAL16R4D 5-29
PAL16R4A 5-37
PAL16R4A-2 5-41
PAL16R4A-4 5-43
PAL16R4B 5-31
PAL16R4B-2 5-35
PAL16R4B-4 5-39
PAL16R6D 5-29
PAL16R6A 5-37
PAL16R6A-2 5-41
PAL16R6A-4 5-43
PAL16R6B 5-31
PAL16R6B-2 5-35
PAL16R6B-4 5-39
PAL16R8D 5-29
PAL16R8A 5-37
PAL16R8A-2 5-41
PAL16R8A-4 5-43
PAL16R8B 5-31
PAL16R8B-2 5-35
PAL16R8B-4 5-39
PAL16RA8 5-11
PAL16RP4A 5-17
PAL16RP6A 5-17
PAL16RP8A 5-17
PAL16X4 5-51

PAL18L4 5-147

PAL20C1 5-147
PAL20L2 5-147
PAL20L10A 5-113
PAL20L8A 5-128
PAL20L8A-2 5-130
PAL20L8B 5-125
PAL20L8B-2 5-126
PAL20R4A 5-128
PAL20R4A-2 5-130
PAL20R4B 5-125
PAL20R4B-2 5-126
PAL20R6A 5-128
PAL20R6A-2 5-130
PAL20R6B 5-125
PAL20R6B-2 5-126
PAL20R8A 5-128
PAL20R8A-2 5-130
PAL20R8B 5-125
PAL20R8B-2 5-126
PAL20RA10 5-97
PAL20RA10-20 5-95
PAL20RS10 5-103
PAL20RS4 5-103
PAL20RS8 5-103
PAL20S10 5-103
PAL20X4A 5-113
PAL20X8A 5-113
PAL20X10A 5-113

PAL22RX8A 5-87

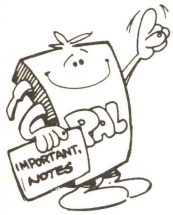
PAL32R16 5-158
PAL32VX10 5-70
PAL32VX10A 5-70

PAL10020EV/EG8 5-381

PALC16L8Q-25 5-33
PALC16L8Z-25 5-50
PALC16R4Q-25 5-33
PALC16R4Z-25 5-50
PALC16R6Q-25 5-33
PALC16R6Z-25 5-50
PALC16R8Q-25 5-33
PALC16R8Z-25 5-50
PALC20L8Z-35 5-133
PALC20R4Z-35 5-133
PALC20R6Z-35 5-133
PALC20R8Z-35 5-133
PALC20L8Z-45 5-133
PALC20R4Z-45 5-133
PALC20R6Z-45 5-133
PALC20R8Z-45 5-133
PALC22V10H-25 5-79
PALC22V10H-35 5-79

PLS105-37 5-331
PLS167-33 5-331
PLS168-33 5-331

PMS14R21 5-315
PMS14R21A 5-315


IMPORTANT.
NOTES

## Data Sheets


TTL/CMOS PAL Devices

TTL/CMOS AmPAL Devices

PROSE/PLS Sequencers

FPC/PEG Sequencers

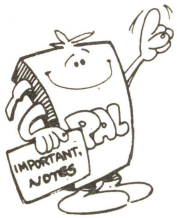
ECL PAL Devices

HAL/ZHAL Devices

Military PAL Devices

Logic Cell Array

Electrical Definitions


IMPORTANT.
NOTES

# PAL/PLD Device Menu

| DEVICE NAME | INPUTS | OUTPUTS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ in ns) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|
| PAL8L14A | 8 | 14 | 1 | 25 | 90 | 5-141 |
| PAL6L16A | 6 | 16 | 1 | 25 | 90 | 5-141 |
| PAL10H8 | 10 | 8 | 2 | 35 | 90 | 5-56 |
| PAL12H6 | 12 | 6 | 2,4 | 35 | 90 | 5-56 |
| PAL14H4 | 14 | 4 | 4 | 35 | 90 | 5-56 |
| PAL16H2 | 16 | 2 | 8 | 35 | 90 | 5-56 |
| PAL10L8 | 10 | 8 | 2 | 35 | 90 | 5-56 |
| PAL12L6 | 12 | 6 | 2,4 | 35 | 90 | 5-56 |
| PAL14L4 | 14 | 4 | 4 | 35 | 90 | 5-56 |
| PAL16L2 | 16 | 2 | 8 | 35 | 90 | 5-56 |
| PAL16C1 | 16 | 2 | 16 | 40 | 90 | 5-56 |
| PAL16L8D | 16* | 8 | 7 | 10 | 180 | 5-29 |
| AmPAL16L8D | | | | 10 | 180 | 5-183 |
| PAL16L8B | | | | 15 | 180 | 5-31 |
| AmPAL16L8B | | | | 15 | 180 | 5-197 |
| PALC16L8Z-25 | | | | 25 | 0.1 | 5-50 |
| PALC16L8Q-25 | | | | 25 | 45 | 5-33 |
| PAL16L8B-2 | | | | 25 | 90 | 5-35 |
| AmPAL16L8AL | | | | 25 | 90 | 5-197 |
| PAL16L8A | | | | 25 | 180 | 5-37 |
| AmPAL16L8A | | | | 25 | 155 | 5-197 |
| PAL16L8B-4 | | | | 35 | 55 | 5-39 |
| AmPAL16L8Q | | | | 35 | 45 | 5-197 |
| PAL16L8A-2 | | | | 35 | 90 | 5-41 |
| AmPAL16L8L | | | | 35 | 80 | 5-197 |
| AmPAL16L8 | | | | 35 | 155 | 5-197 |
| PAL16L8A–4 | | | | 55 | 50 | 5-43 |
| PAL16P8A | 16* | 8 | 7 | 25/30** | 180 | 5-17 |
| AmPAL18P8B | 18* | 8 | 8 | 15 | 180 | 5-202 |
| AmPAL18P8AL | | | | 25 | 90 | 5-202 |
| AmPAL18P8A | | | | 25 | 180 | 5-202 |
| AmPAL18P8Q | | | | 35 | 55 | 5-202 |
| AmPAL18P8L | | | | 35 | 90 | 5-202 |
| PAL12L10 | 12 | 10 | 2 | 40 | 100 | 5-147 |
| PAL14L8 | 14 | 8 | 2,4 | 40 | 100 | 5-147 |
| PAL16L6 | 16 | 6 | 2,4 | 40 | 100 | 5-147 |
| PAL18L4 | 18 | 4 | 4,6 | 40 | 100 | 5-147 |
| PAL20L2 | 20 | 2 | 8 | 40 | 100 | 5-147 |
| PAL20C1 | 20 | 2 | 16 | 40 | 100 | 5-147 |
| PAL20L8B | 20* | 8 | 7 | 15 | 210 | 5-125 |
| PAL20L8B-2 | | | | 25 | 105 | 5-126 |
| PAL20L8A | | | | 25 | 210 | 5-128 |
| PALC20L8Z-35 | | | | 35 | 0.1 | 5-133 |
| PAL20L8A-2 | | | | 35 | 105 | 5-130 |
| PALC20L8Z-45 | | | | 45 | 0.1 | 5-133 |

Table 1. Simple Combinatorial PAL Devices

| DEVICE NAME | INPUTS | OUTPUTS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ in ns) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|
| AmPAL20L10B | 20* | 10 | 3 | 15 | 210 | 5-306 |
| AmPAL20L10-20 | | | | 20 | 165 | 5-306 |
| AmPAL20L10AL | | | | 25 | 105 | 5-306 |
| PAL20L10A | | | | 30 | 165 | 5-113 |
| PAL20S10 | 20* | 10 | 0–16†† | 35 | 240 | 5-103 |
| AmPAL22P10B | 22* | 10 | 8 | 15 | 210 | 5-306 |
| AmPAL22P10AL | | | | 25 | 105 | 5-306 |
| AmPAL22P10A | | | | 25 | 210 | 5-306 |
| AmPAL22XP10-20 | 22* | 10 | 2/6† | 20 | 210 | 5-286 |
| AmPAL22XP10-30L | | | | 30 | 105 | 5-286 |
| AmPAL22XP10-30 | | | | 30 | 180 | 5-286 |
| AmPAL22XP10-40L | | | | 40 | 105 | 5-286 |

* Includes feedback
** Depending on polarity
† Has an exclusive-OR gate
†† Product term steering

**Table 1. Simple Combinatorial PAL Devices** (Cont'd.)

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PAL16R8D | 16* | 8 | 8 | 8 | 55 | 180 | 5-29 |
| AmPAL16R8D | | | | | 55 | 180 | 5-183 |
| PAL16R8B | | | | | 37 | 180 | 5-31 |
| AmPAL16R8B | | | | | 40 | 180 | 5-197 |
| PALC16R8Z-25 | | | | | 28.5 | 0.1 | 5-50 |
| PALC16R8Q-25 | | | | | 28.5 | 45 | 5-33 |
| PAL16R8B-2 | | | | | 25 | 90 | 5-35 |
| AmPAL16R8AL | | | | | 28.5 | 90 | 5-197 |
| PAL16R8A | | | | | 25 | 180 | 5-37 |
| AmPAL16R8A | | | | | 28.5 | 155 | 5-197 |
| PAL16R8B-4 | | | | | 16 | 55 | 5-39 |
| AmPAL16R8Q | | | | | 18 | 45 | 5-197 |
| PAL16R8A-2 | | | | | 16 | 90 | 5-41 |
| AmPAL16R8L | | | | | 18 | 80 | 5-197 |
| AmPAL16R8 | | | | | 18 | 155 | 5-197 |
| PAL16R8A-4 | | | | | 11 | 50 | 5-43 |
| PAL16R6D | 16* | 8 | 6 | 8 | 55 | 180 | 5-29 |
| AmPAL16R6D | | | | | 55 | 180 | 5-183 |
| PAL16R6B | | | | | 37 | 180 | 5-31 |
| AmPAL16R6B | | | | | 40 | 180 | 5-197 |
| PALC16R6Z-25 | | | | | 28.5 | 0.1 | 5-50 |
| PALC16R6Q-25 | | | | | 28.5 | 45 | 5-33 |
| PAL16R6B-2 | | | | | 25 | 90 | 5-35 |
| AmPAL16R6AL | | | | | 28.5 | 90 | 5-197 |
| PAL16R6A | | | | | 25 | 180 | 5-37 |

**Table 2. Simple Registered PAL Devices**

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| AmPAL16R6A | | | | | 28.5 | 180 | 5-197 |
| PAL16R6B-4 | | | | | 16 | 55 | 5-39 |
| AmPAL16R6Q | | | | | 18 | 45 | 5-197 |
| PAL16R6A-2 | | | | | 16 | 90 | 5-41 |
| AmPAL16R6L | | | | | 18 | 90 | 5-197 |
| AmPAL16R6 | | | | | 18 | 180 | 5-197 |
| PAL16R6A-4 | | | | | 11 | 50 | 5-43 |
| PAL16R4D | 16* | 8 | 4 | 8 | 55 | 180 | 5-29 |
| AmPAL16R4D | | | | | 55 | 180 | 5-183 |
| PAL16R4B | | | | | 37 | 180 | 5-31 |
| AmPAL16R4B | | | | | 40 | 180 | 5-197 |
| PALC16R4Z-25 | | | | | 28.5 | 0.1 | 5-50 |
| PALC16R4Q-25 | | | | | 28.5 | 45 | 5-33 |
| PAL16R4B-2 | | | | | 25 | 90 | 5-35 |
| AmPAL16R4AL | | | | | 28.5 | 90 | 5-197 |
| PAL16R4A | | | | | 25 | 180 | 5-37 |
| AmPAL16R4A | | | | | 28.5 | 180 | 5-197 |
| PAL16R4B-4 | | | | | 16 | 55 | 5-39 |
| AmPAL16R4Q | | | | | 18 | 45 | 5-197 |
| PAL16R4A-2 | | | | | 16 | 90 | 5-41 |
| AmPAL16R4L | | | | | 18 | 90 | 5-197 |
| AmPAL16R4 | | | | | 18 | 180 | 5-197 |
| PAL16R4A-4 | | | | | 11 | 50 | 5-43 |
| PAL16X4 | 16* | 8 | 4 | 8† | 14 | 225 | 5-51 |
| PAL16RP8A | 16* | 8 | 8 | 8 | 25** | 180 | 5-17 |
| PAL16RP6A | 16* | 8 | 6 | 8 | 25** | 180 | 5-17 |
| PAL16RP4A | 16* | 8 | 4 | 8 | 25** | 180 | 5-17 |
| PAL20R8B | 20* | 8 | 8 | 8 | 37 | 210 | 5-125 |
| PAL20R8B-2 | | | | | 25 | 105 | 5-126 |
| PAL20R8A | | | | | 25 | 210 | 5-128 |
| PALC20R8Z-35 | | | | | 20 | 0.1 | 5-133 |
| PAL20R8A-2 | | | | | 16 | 105 | 5-130 |
| PALC20R8Z-45 | | | | | 15.3 | 0.1 | 5-133 |
| PAL20R6B | 20* | 8 | 6 | 8 | 37 | 210 | 5-125 |
| PAL20R6B-2 | | | | | 25 | 105 | 5-126 |
| PAL20R6A | | | | | 25 | 210 | 5-128 |
| PALC20R6Z-35 | | | | | 20 | 0.1 | 5-133 |
| PAL20R6A-2 | | | | | 16 | 105 | 5-130 |
| PALC20R6Z-45 | | | | | 15.3 | 0.1 | 5-133 |
| PAL20R4B | 20* | 8 | 4 | 8 | 37 | 210 | 5-125 |
| PAL20R4B-2 | | | | | 25 | 105 | 5-126 |
| PAL20R4A | | | | | 25 | 210 | 5-128 |
| PALC20R4Z-35 | | | | | 20 | 0.1 | 5-133 |
| PAL20R4A-2 | | | | | 16 | 105 | 5-130 |
| PALC20R4Z-45 | | | | | 15.3 | 0.1 | 5-133 |

**Table 2. Simple Registered PAL Devices** (Cont'd.)

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PAL20RS10 | 20* | 10 | 10 | 0–16†† | 20 | 240 | 5-103 |
| PAL20RS8 | 20* | 10 | 8 | 0–16†† | 20 | 240 | 5-103 |
| PAL20RS4 | 20* | 10 | 4 | 0–16†† | 20 | 240 | 5-103 |
| AmPAL22V10-15 | 22* | 10 | 0–10§ | 8–16§§ | 40 | 180 | 5-249 |
| PALC22V10H-25 | | | | | 33.3 | 90 | 5-79 |
| AmPAL22V10A | | | | | 28.5 | 180 | 5-260 |
| PALC22V10H-35 | | | | | 20 | 90 | 5-79 |
| AmPAL22V10 | | | | | 18 | 180 | 5-260 |
| AmPAL20RP10B | 22* | 10 | 10 | 8 | 37 | 210 | 5-306 |
| AmPAL20RP10AL | | | | | 25 | 105 | 5-306 |
| AmPAL20RP10A | | | | | 25 | 210 | 5-306 |
| AmPAL20RP8B | 22* | 10 | 8 | 8 | 37 | 210 | 5-306 |
| AmPAL20RP8AL | | | | | 25 | 105 | 5-306 |
| AmPAL20RP8A | | | | | 25 | 210 | 5-306 |
| AmPAL20RP6B | 22* | 10 | 6 | 8 | 37 | 210 | 5-306 |
| AmPAL20RP6AL | | | | | 25 | 105 | 5-306 |
| AmPAL20RP6A | | | | | 25 | 210 | 5-306 |
| AmPAL20RP4B | 22* | 10 | 4 | 8 | 37 | 210 | 5-306 |
| AmPAL20RP4AL | | | | | 25 | 105 | 5-306 |
| AmPAL20RP4A | | | | | 25 | 210 | 5-306 |
| PAL32R16 | 32* | 16 | 16§ | 0–16†† | 16 | 280 | 5-158 |

* Includes feedback
** With polarity fuse intact
† Has an exclusive-OR gate
†† Product term steering
§ Flip-flops can be bypassed
§§ Has varied product term distribution

**Table 2. Simple Registered PAL Devices** (Cont'd.)

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | FLIP-FLOP TYPES | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STAND BY $I_{CC}$ (mA) | DATASHEET PAGE NO. |
|---|---|---|---|---|---|---|---|---|
| PAL20X10A | 20* | 10 | 10 | D,T,JK,SR | 2/2† | 22.2 | 180 | 5-113 |
| PAL20X8A | 20* | 10 | 8 | D,T,JK,SR | 2/2† | 22.2 | 180 | 5-113 |
| PAL20X4A | 20* | 10 | 4 | D,T,JK,SR | 2/2† | 22.2 | 180 | 5-113 |
| AmPAL20XRP10-20 | 22* | 10 | 10 | D,T,JK,SR | 2/6† | 30.3 | 210 | 5-286 |
| AmPAL20XRP10-30L | | | | | | 22.2 | 105 | 5-286 |
| AmPAL20XRP10-30 | | | | | | 22.2 | 180 | 5-286 |
| AmPAL20XRP10-40L | | | | | | 14.3 | 105 | 5-286 |
| AmPAL20XRP8-20 | 22* | 10 | 8 | D,T,JK,SR | 2/6, 8† | 30.3 | 210 | 5-286 |
| AmPAL20XRP8-30L | | | | | | 22.2 | 105 | 5-286 |
| AmPAL20XRP8-30 | | | | | | 22.2 | 180 | 5-286 |
| AmPAL20XRP8-40L | | | | | | 14.3 | 105 | 5-286 |
| AmPAL20XRP6-20 | 22* | 10 | 6 | D,T,JK,SR | 2/6, 8† | 30.3 | 210 | 5-286 |
| AmPAL20XRP6-30L | | | | | | 22.2 | 105 | 5-286 |
| AmPAL20XRP6-30 | | | | | | 22.2 | 180 | 5-286 |
| AmPAL20XRP6-40L | | | | | | 14.3 | 105 | 5-286 |
| AmPAL20XRP4-20 | 22* | 10 | 4 | D,T,JK,SR | 2/6, 8† | 30.3 | 210 | 5-286 |
| AmPAL20XRP4-30L | | | | | | 22.2 | 105 | 5-286 |
| AmPAL20XRP4-30 | | | | | | 22.2 | 180 | 5-286 |
| AmPAL20XRP4-40L | | | | | | 14.3 | 105 | 5-286 |
| PAL22RX8A | 22* | 8 | 8§ | D,T,JK,SR | 1/8† | 28.5 | 210 | 5-87 |
| AmPAL23S8-20 | 23* | 8 | 14§ | D,B◊ | 6–12§§ | 33 | 200 | 5-169 |
| AmPAL23S8-25 | | | | | | 28.5 | 200 | 5-169 |
| AmPALC29M16-35 | 29* | 16 | 16§ | D,B,L◊ | 8–16§§ | 20 | 120 | 5-231 |
| AmPALC29M16-45 | | | | | | 15 | 120 | 5-231 |
| PAL32VX10A | 32* | 10 | 10§ | D,T,JK,SR, B◊ | 1/8–16† | 25 | 180 | 5-70 |
| PAL32VX10 | | | | | | 22.2 | 180 | 5-70 |

* Includes feedback
† Has an exclusive-OR gate
§ Some flip-flops can be bypassed
§§ Has varied product term distribution
◊ B=flip-flops are or can be buried; L=latched outputs possible

Table 3. State Machine PAL Devices

| DEVICE NAME | INPUTS | OUTPUTS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ in ns) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|
| PAL16RA8 | 16* | 8 | 4 | 30** | 170 | 5-11 |
| PAL20RA10-20 | 20* | 10 | 4 | 20** | 200 | 5-95 |
| PAL20RA10 | | | | 30** | 200 | 5-97 |
| AmPALC29MA16-35 | 29* | 16 | 4–12†† | 35 | 120 | 5-209 |
| AmPALC29MA16-45 | | | | 45 | 120 | 5-209 |

* Includes feedback
** With polarity fuse intact
†† Has product term steering

Table 4. Asynchronous PAL Devices

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ or $f_{MAX}$) | $I_{EE}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PAL10H20P8 | 20* | 8 | 0 | 0–8†† | 6 ns | 210 | 5-386 |
| PAL10H20G8 | 20* | 8 | 8§ | 0–8†† | 6 ns | 225 | 5-382 |
| PAL10H20EV/EG8 | 20* | 8 | 8§ | 8–12§§ | 125 MHz | 220 | 5-381 |

* Includes feedback
§ Flip-flops can be bypassed
†† Has product term steering
§§ Has varied product term distribution

**Table 5. 10KH-Compatible PAL Devices**

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ or $f_{MAX}$) | $I_{EE}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PAL10020EV/EG8 | 20* | 8 | 8§ | 8–12§§ | 125 MHz | 220 | 5-381 |

* Includes feedback
§ Flip-flops can be bypassed
§§ Has varied product term distribution

**Table 6. 100K-Compatible PAL Devices**

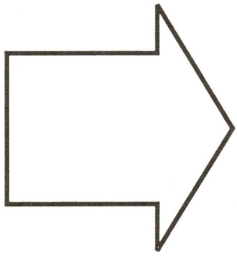
| DEVICE NAME | INPUTS | OUTPUTS | STATES (MAX) | BRANCHES PER STATE | SPEED ($f_{MAX}$ in MHz) | $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PMS14R21A | 8 | 8 | 128 | 4 | 30 | 210 | 5-315 |
| PMS14R21 | | | | | 25 | 210 | 5-315 |
| PLS105-37 | 16 | 8 | <64* | * | 37 | 200 | 5-331 |
| PLS167-33 | 14 | 6 | <128* | * | 33 | 200 | 5-331 |
| PLS168-33 | 12 | 8 | <1028* | * | 33 | 200 | 5-331 |
| Am29PL141 | 6 | 16 | 64 | 2 | 20 | 450 | 5-339 |
| Am2971 | 6 | 14 | N/A | N/A | 85 | 310 | 5-365 |

* Depends highly on state diagram topology. May be limited by number of product terms or number of flip-flops.

**Table 7. Programmable Sequencers**

| DEVICE NAME | I/O PINS | CLBs | SPEED (INTERNAL TOGGLE $f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|
| M2064-70 | 58 | 64 | 70 | 5 | 5-483 |
| M2064-50 | | | 50 | 5 | 5-483 |
| M2064-33 | | | 33 | 5 | 5-483 |
| M2018-50 | 74 | 100 | 50 | 5 | 5-483 |
| M2018-33 | | | 33 | 5 | 5-483 |

**Table 8. LCA Devices**



## Data Sheets


TTL/CMOS PAL Devices

TTL/CMOS AmPAL Devices

PROSE/PLS Sequencers

FPC/PEG Sequencers

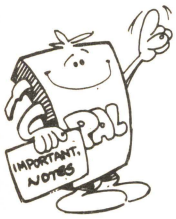
ECL PAL Devices

HAL/ZHAL Devices

Military PAL Devices

Logic Cell Array

Electrical Definitions

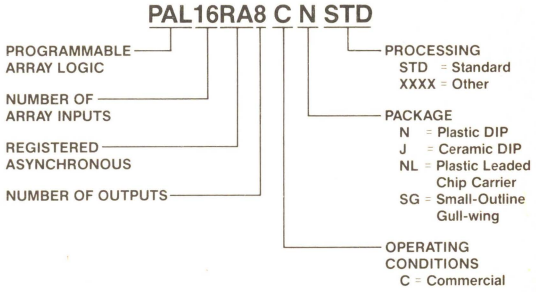

IMPORTANT
NOTES

# Asynchronous PAL16RA8

## Features/Benefits

- Programmable clock for asynchronous operation
- Programmable asynchronous set and reset
- Programmable polarity
- Programmable flip-flop bypass
- Local and global output enable control
- TTL level register preload
- Power-up reset
- Complements 24-pin PAL20RA10
- High speed, at 30 ns $t_{PD}$
- Security fuse

## Ordering Information




## Description

The PAL16RA8 is a 20-pin registered version of the original asynchronous PAL device, the PAL20RA10. This versatile device features programmable clock, enable, set, and reset, all of which can operate asynchronously to other flip-flops in the same device. It also has individual flip-flop bypass, allowing this one device to provide any combination of registered and combinatorial outputs.

## Programmable Clock

The clock input to each flip-flop comes from the programmable array, allowing the flip-flops to be clocked independently if desired.

## Programmable Set and Reset

Each flip-flop has a product line for asynchronous set and one product for asynchronous reset. If the chosen product line is high, the flip-flop will set (become a logic high) or reset (become a logic low). The sense of the output pin is inverted since the output is active low.

## Programmable Polarity

Each flip-flop has individually programmable polarity. The unprogrammed state is active low.

## Programmable Flip-Flop Bypass

If both the set and reset product lines are high, the flip-flop is bypassed, and the output becomes combinatorial. Thus each output can be configured to be registered or combinatorial.

## Programmable and Hard-Wired Three-State Outputs

The PAL16RA8 provides a product term dedicated to output control. There is also an output control pin (pin 11). The output is enabled if both the output control pin is low and the output control product term is high. If the output control pin is high, all outputs will be disabled. If an output control product term is low, then that output will be disabled.

## Register Preload and Power-Up Reset

Each device also offers register preload for device testability. The registers can be preloaded from the Outputs by using TTL level signals in order to simplify functional testing. This series also offers Power-Up Reset, whereby the registers power up to a logic low, setting the active-low outputs to a logic high.

## Packages

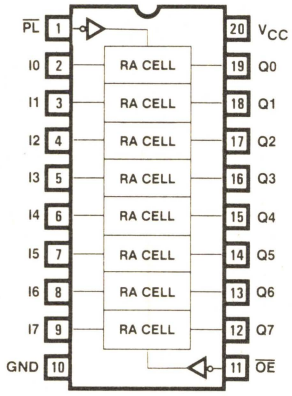
The PAL16RA8 is available in the plastic DIP (N), ceramic DIP (J), plastic leaded chip carrier (NL), and small outline (SG) packages.



10232A
JANUARY 1988

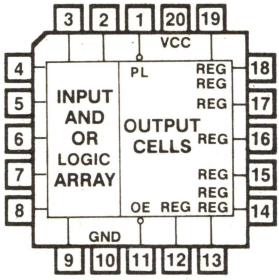
## DIP/SO Pinout

16RA8




## PLCC Pinout

16RA8




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | | 20 | 13 | | ns |
| $t_{wp}$ | Preload pulse width | | 35 | 15 | | ns |
| $t_{su}$ | Set up time from input or feedback to clock | | 20 | 10 | | ns |
| $t_{sup}$ | Preload set up time | | 25 | 5 | | ns |
| $t_h$ | Hold time | Polarity fuse intact | 10 | −2 | | ns |
| | | Polarity fuse programmed | 0 | −6 | | |
| $t_{hp}$ | Preload hold time | | 25 | 5 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 135 | 170 | mA |

1. The PAL16RA8 Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | Polarity fuse intact | $R_1$ = 560 Ω<br>$R_2$ = 1.1K Ω | | 20 | 30 | ns |
| | | Polarity fuse programmed | | | 25 | 35 | |
| $t_{CLK}$ | Clock to output or feedback | | | 10 | 17 | 30 | ns |
| $t_S$ | Input to asynchronous set | | | | 22 | 35 | ns |
| $t_R$ | Input to asynchronous reset | | | | 27 | 40 | ns |
| $t_{PZX}$ | Pin 11 to output enable | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 11 to output disable | | | | 10 | 20 | ns |
| $t_{EA}$ | Input to output enable | | | | 18 | 30 | ns |
| $t_{ER}$ | Input to output disable | | | | 15 | 30 | ns |
| $f_{MAX}$ | Maximum frequency | External | | 20 | 35 | | MHz |
| | | No feedback | | 25 | 38 | | |

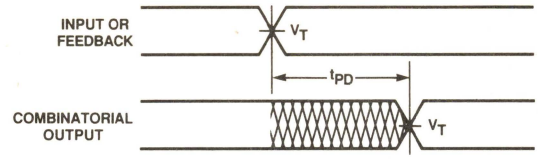
### Switching Test Load
(refer to page 5-164)

### Power-Up Reset Waveform
(refer to page 5-164)

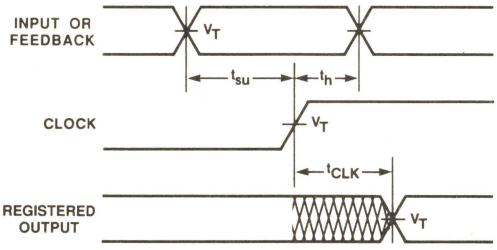
### Programmers/Development Systems
(refer to Programmer Reference Guide, page 3-81)

### Schematic of Inputs and Outputs
(refer to page 5-164)

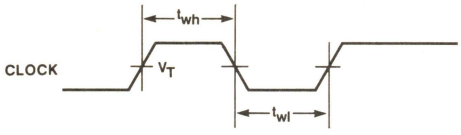
## Switching Waveforms



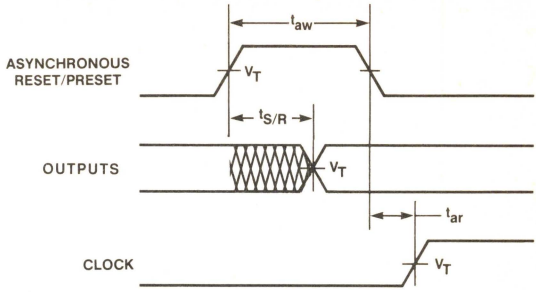

**Combinatorial Output**



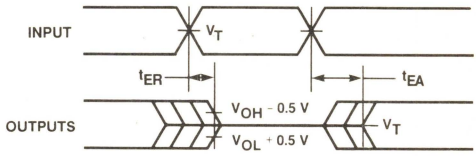

**Registered Output**



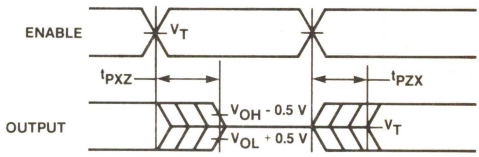

**Clock Width**




**Asynchronous Reset/Preset**




**Input Output Disable/Enable**




**Pin 11 to Output Disable/Enable**

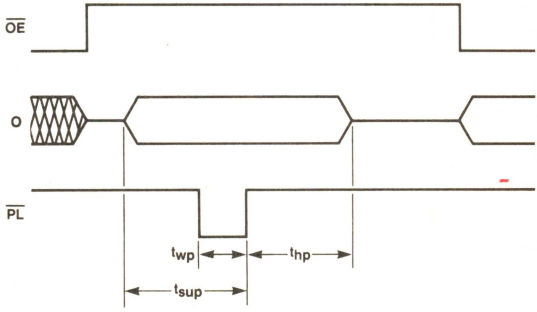
Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagrams

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE; CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

## Register Preload

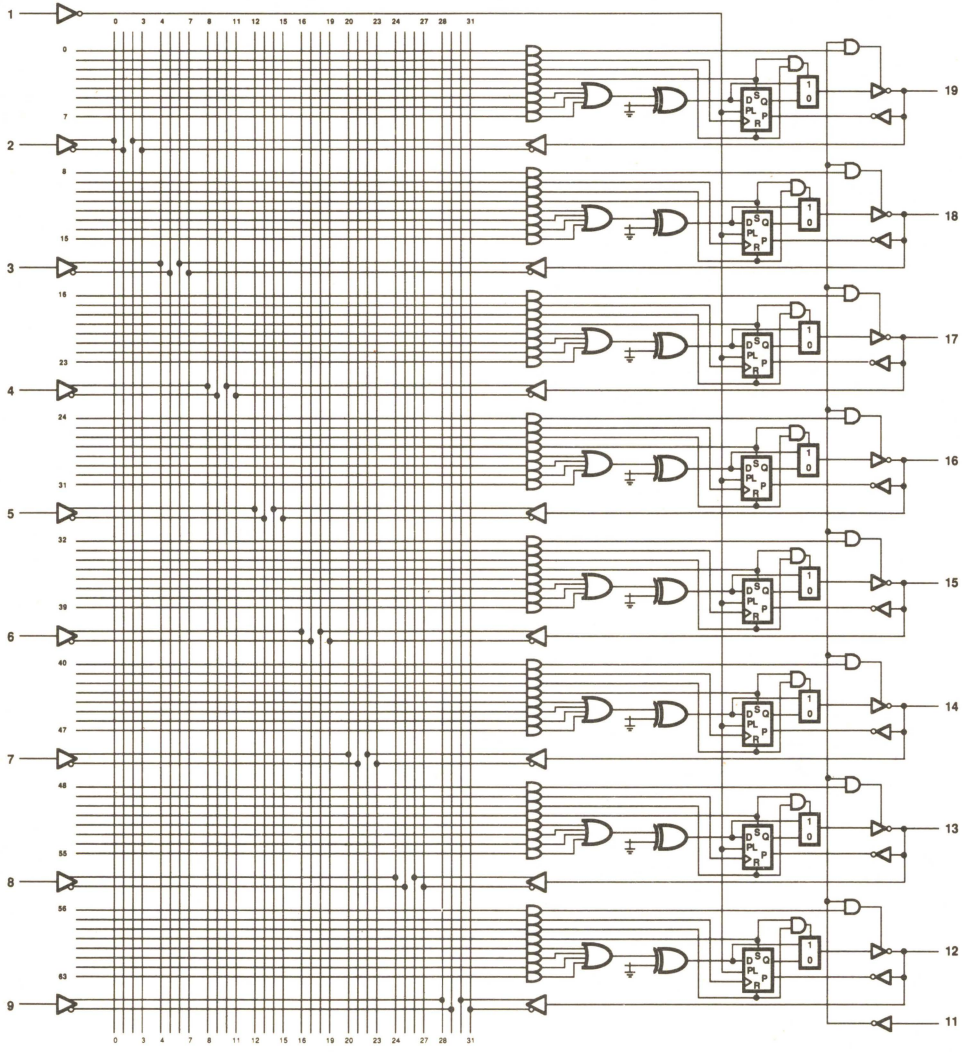
Register preload allows any arbitrary state to be loaded into the PAL device output registers. This allows complete logic verification, including states that are impossible or impractical to reach. To use the preload feature, first disable the outputs by bringing $\overline{OE}$ high, and present the data at the output pins. A low level on the preload pin ($\overline{PL}$) will then load the data into the registers.




5

## Logic Diagram

PAL16RA8



# PAL16RP8A Series

# 16P8A, 16RP8A 16RP6A, 16RP4A

## Features/Benefits

- Programmable polarity
- High speed at 25 ns tPD
- Register preload
- Power-up reset
- Security fuse

## Description

The PAL16RP8A Series is equivalent to the PAL16R8A Series, with the addition of programmable polarity. With programmable polarity unused, these devices are equivalent to the PAL16R8A Series.

## Variable Input/Output Pin Ratio

The registered devices have eight dedicated input lines, and each combinatorial output is an I/O pin. The combinatorial device has ten dedicated input lines, and only six of the eight combinatorial outputs are I/O pins. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

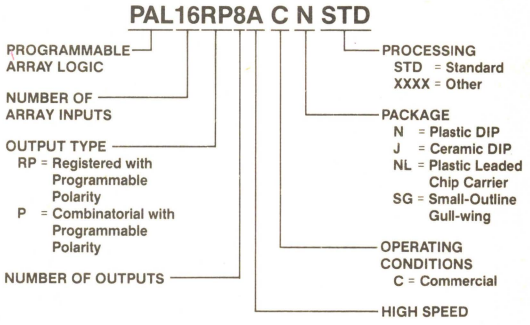
## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Ordering Information




## Polarity

Each of these devices offers programmable polarity on each output. If the polarity fuse is unused, the output is active low. If the polarity fuse is programmed, the output is inverted to active high.

## Preload and Power-Up Reset

Each device also offers register preload for device testability. The registers can be preloaded from the outputs by using supervoltages in order to simplify functional testing. This series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

## Performance

Performance varies according to the use of the programmable polarity. Active low outputs have a tPD of 25 ns, while active high outputs have a tPD of 30 ns due to the extra inversion. All devices consume 180 mA maximum ICC.

## Packages

The commercial PAL16RP8A Series is available in the plastic DIP (N), ceramic DIP (J), plastic leaded chip carrier (NL), and small outline (SG) packages.

## PAL16RP8A Series

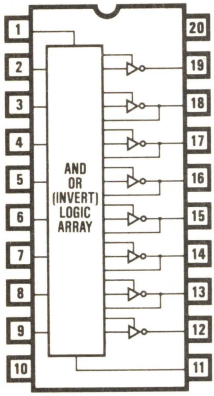
| | ARRAY INPUTS | OUTPUTS | | $t_{PD}$* (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|---|
| | | COMBINATORIAL | REGISTERED | | |
| PAL16P8A | 16 | 8 | 0 | 25/30 | 180 |
| PAL16RP8A | 16 | 0 | 8 | 25/30 | 180 |
| PAL16RP6A | 16 | 6 | 2 | 25/30 | 180 |
| PAL16RP4A | 16 | 4 | 4 | 25/30 | 180 |

*25 ns active low, 30 ns active high

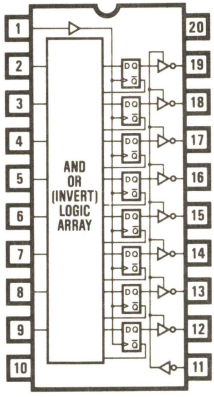
10236A
JANUARY 1988

## DIP/SO Pinouts

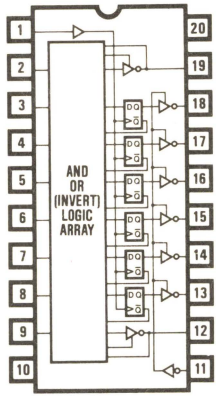
16P8



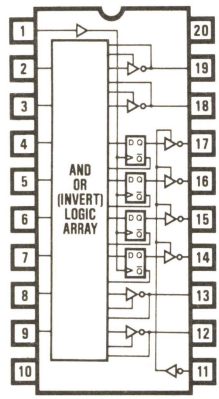

16RP8




16RP6

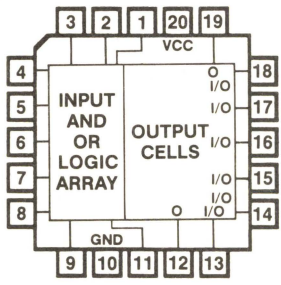



16RP4

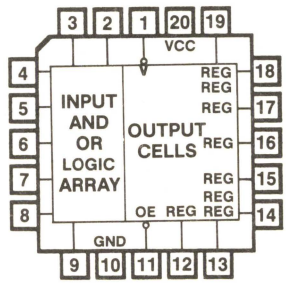



## PLCC Pinouts

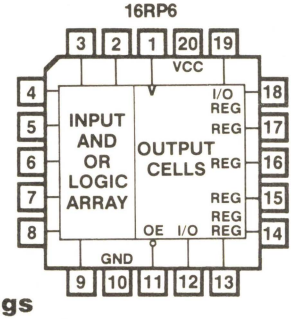
16P8



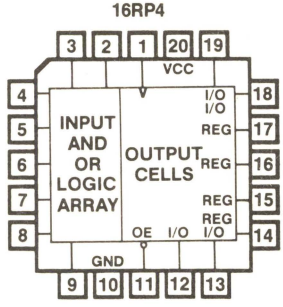

16RP8




16RP6




16RP4




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65° C to +150° C | |

## Operating Conditions

| SYMBOL | PARAMETER | | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | | 20 | 14 | | ns |
| | | High | | 10 | 6 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16RP8A 16RP6A 16RP4A | Polarity fuse intact | 25 | 15 | | ns |
| | | | Polarity fuse programmed | 30 | 20 | | |
| $t_h$ | Hold time | | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | | 0 | | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 120 | 180 | mA |

1. The PAL16RP8A Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 16P8A, 16RP6A, 16RP4A | | Polarity fuse intact | $R_1$ = 200 Ω<br>$R_2$ = 390 KΩ | | 15 | 25 | ns |
| | | | Polarity fuse programmed | | | 20 | 30 | |
| $t_{CLK}$ | Clock to output | | | | | 10 | 15 | ns |
| $t_{CF}$ | Clock to feedback | | | | | 8 | 10 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16P8A | | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16P8A | | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 16P8A, 16RP6A, 16RP4A | | | 10 | 25 | ns |
| $t_{ER}$ | Input to output disable | | 16P8A, 16RP6A, 16RP4A | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency 16RP8A, 16RP6A, 16RP4A | External | Polarity fuse intact | | 25 | 40 | | MHz |
| | | | Polarity fuse programmed | | 22 | 33 | | |
| | | Internal | Polarity fuse intact | | 28.5 | 43 | | |
| | | | Polarity fuse programmed | | 25 | 35 | | |
| | | No feedback | | | 33 | 50 | | |

### Switching Test Load
(refer to page 5-164)

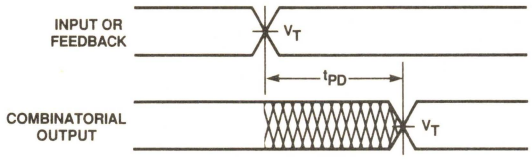
### Programmers/Development Systems
(refer to Programmer Reference Guide, page 3-81)

### Register Preload Waveform
(refer to page 5-164)

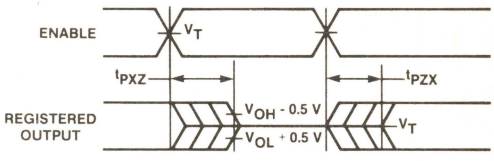
### Power-Up Reset Waveform
(refer to page 5-164)

### Schematic of Inputs and Outputs
(refer to page 5-164)

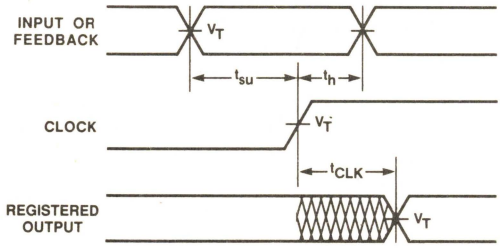
## Switching Waveforms



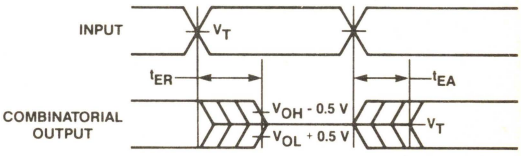

Combinatorial Output



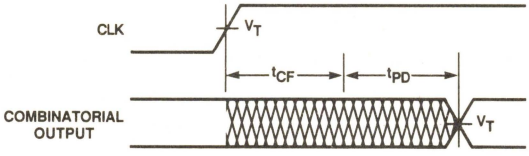

Pin 11 to Output Disable/Enable



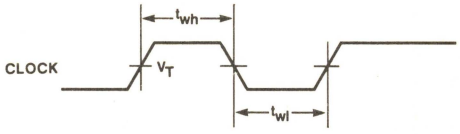

Registered Output



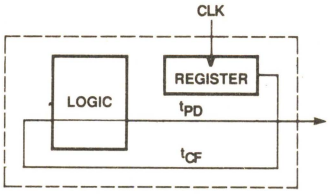

Input to Output Disable/Enable




Clock to Feedback to Combinatorial Output (see path below)




Clock Width




## Key to Timing Diagrams



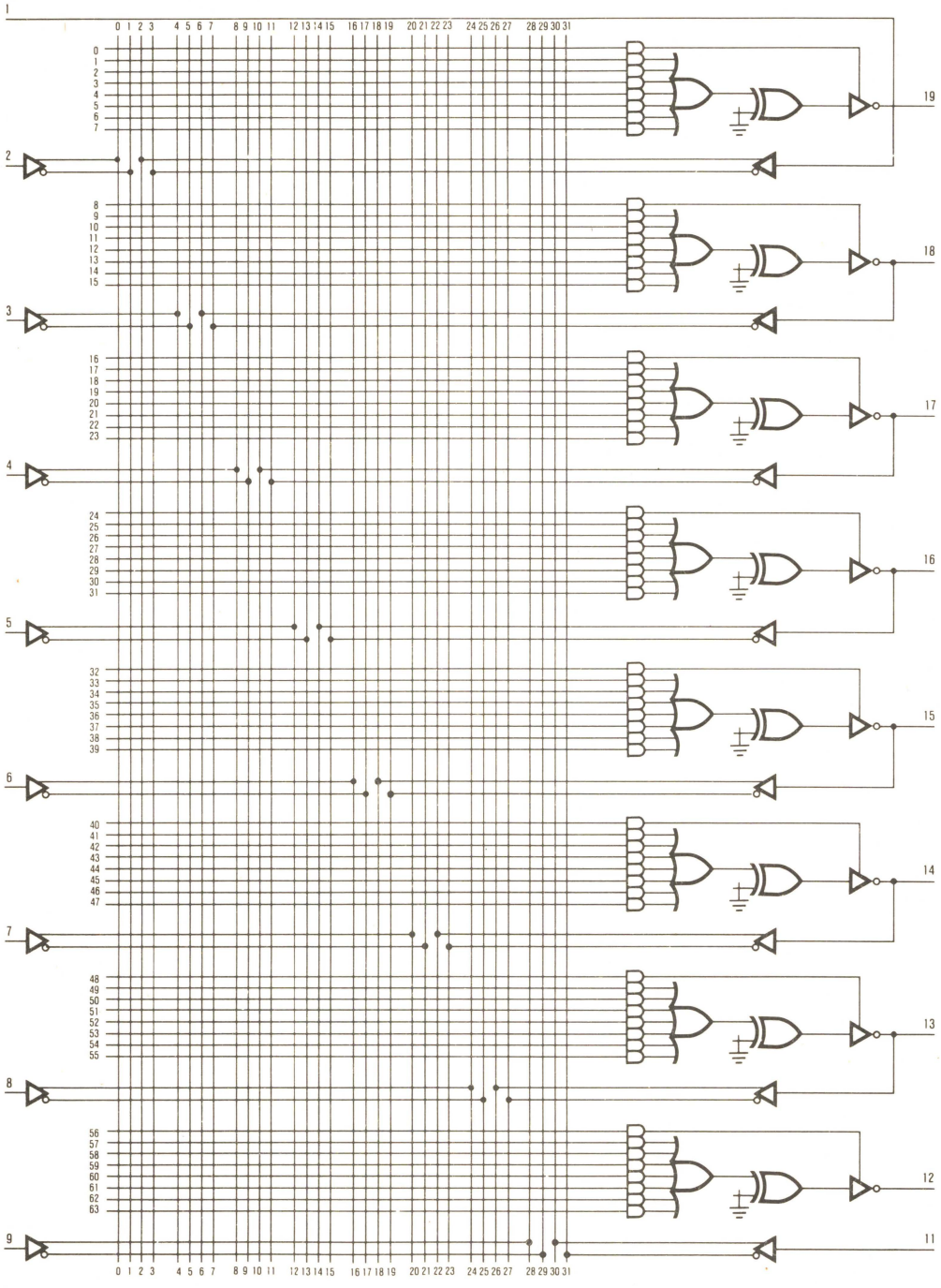
| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE; CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

Notes:
1. VT=1.5V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

5

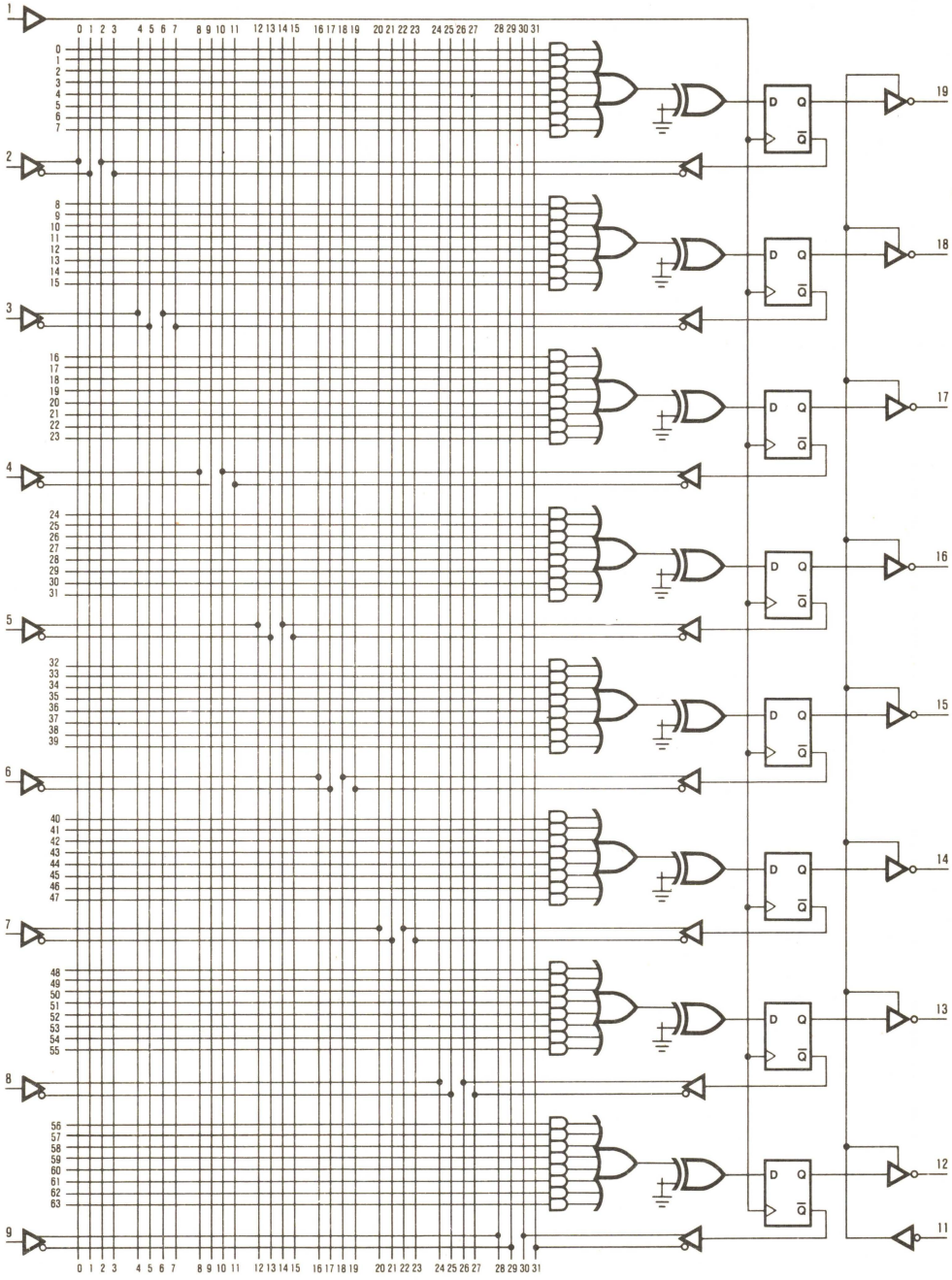
## Logic Diagram

16P8A



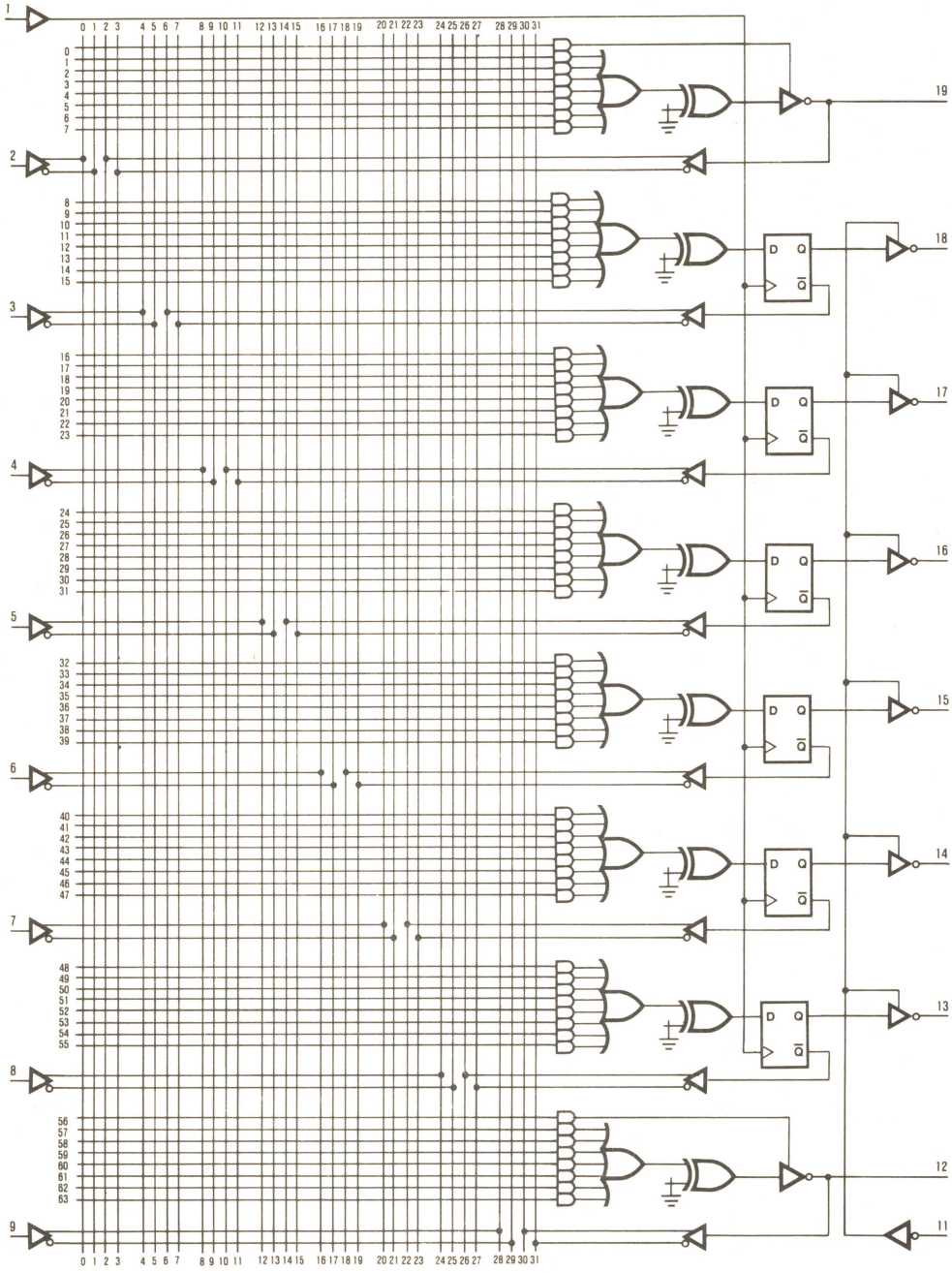
## Logic Diagram

16RP8A



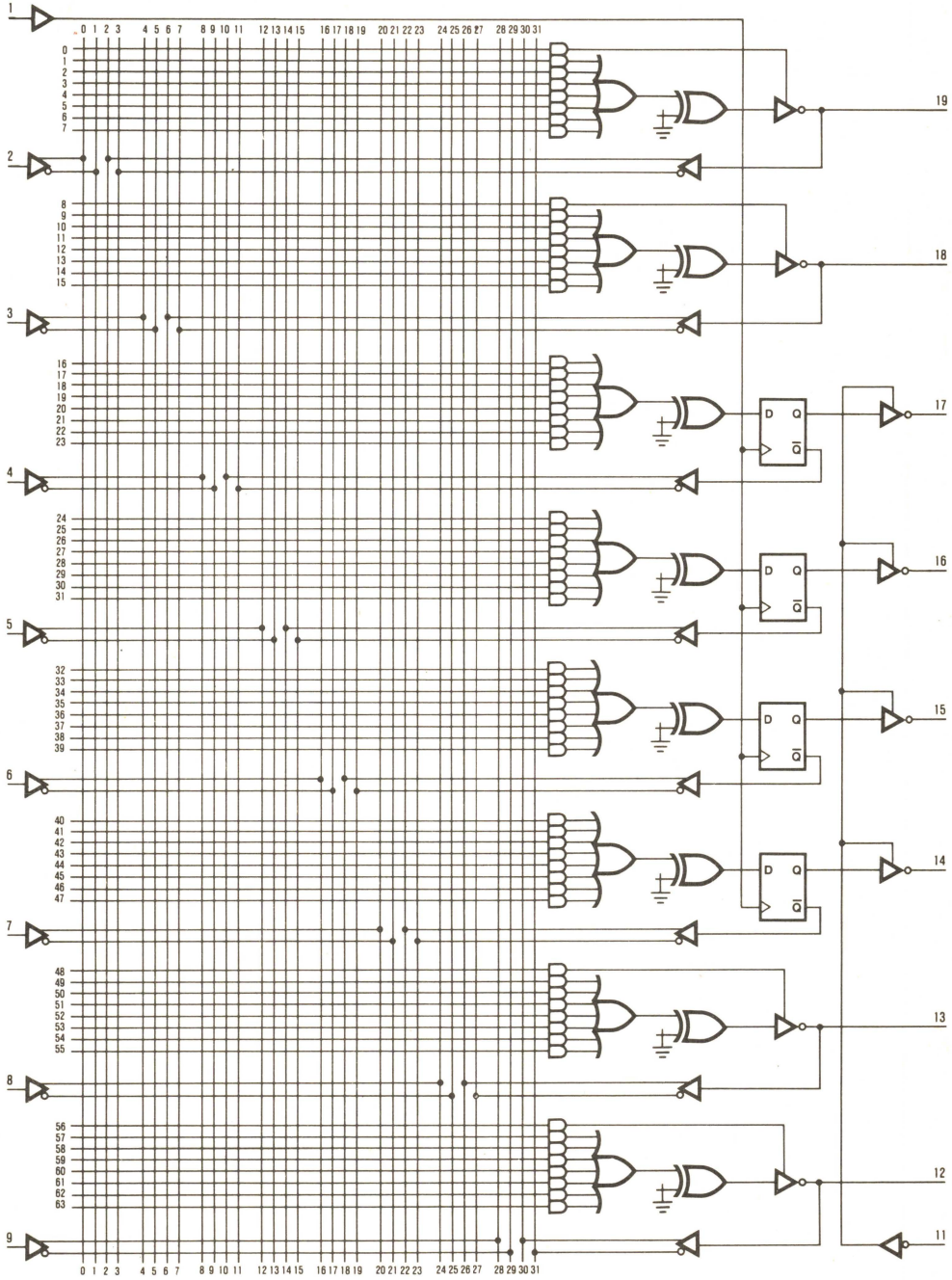
## Logic Diagram

16RP6A



## Logic Diagram

16RP4A



# PAL16R8 Family

# 16L8, 16R8
# 16R6, 16R4

## Features/Benefits

- Standard 20-pin architectures
- TTL and CMOS versions
- High speed, as fast as 10 ns tPD for PAL16R8D Series
- Low power, as low as zero standby for PALC16R8Z Series
- Security fuse/cell on all devices

## Description

The PAL16R8 Series offers the four most popular PAL device architectures. It also provides the fastest PAL devices in the industry.

The PAL16R8 Series consists of four devices, each with sixteen array inputs and eight outputs. The devices have either 0, 4, 6, or 8 registered outputs, with the remaining being combinatorial.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

The registered devices in the series have eight dedicated input lines, and each combinatorial output is an I/O pin. The combinatorial device has ten dedicated input lines, and only six of the eight combinatorial outputs are I/O pins. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

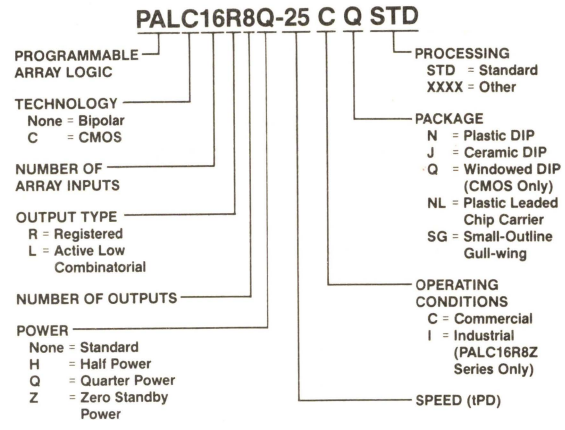
## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

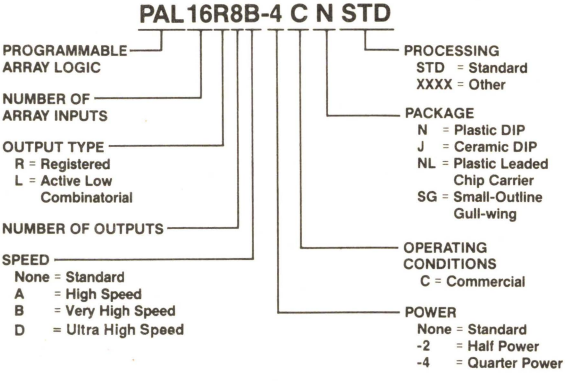
## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Ordering Information — Newer Products




## Ordering Information — Older Products




## Packages

The commercial PAL16R8 Series is available in the plastic DIP (N), ceramic DIP (J), plastic leaded chip carrier (NL), and small outline (SG) packages. The CMOS versions are also available in windowed (Q) packages.

10240A
JANUARY 1988

## Polarity

All outputs are active low.

## Performance

Several speed/power versions are available (see table). The D Series offers the fastest TTL programmable logic devices in the industry at 10 ns tPD.

| | DEDICATED INPUTS | OUTPUTS | |
|---|---|---|---|
| | | COMBINATORIAL | REGISTERED |
| PAL16L8 | 10 | 8 (6 I/O) | 0 |
| PAL16R8 | 8 | 0 | 8 |
| PAL16R6 | 8 | 2 I/O | 6 |
| PAL16R4 | 8 | 4 I/O | 4 |

## Preload

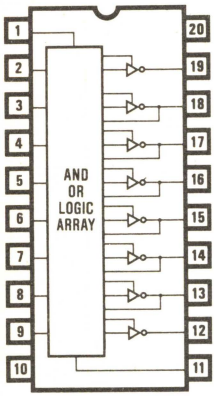
The CMOS Series offers register preload for device testability. The register can be preloaded from outputs by using super-voltages in order to simplify functional testing.

| SUFFIX | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|
| A | 25 | 180 |
| A-2 | 35 | 90 |
| A-4 | 55 | 50 |
| B | 15 | 180 |
| B-2 | 25 | 90 |
| B-4 | 35 | 55 |
| (C)Q-25 | 25 | 45 |
| D | 10 | 180 |

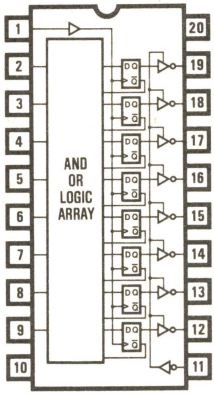
5

## DIP/SO Pinouts

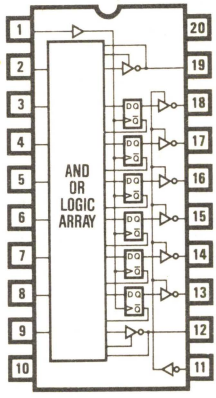
16L8



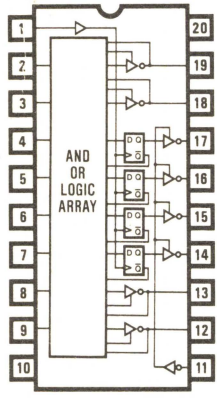

16R8




16R6

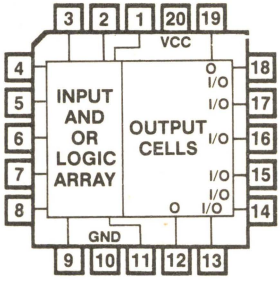



16R4

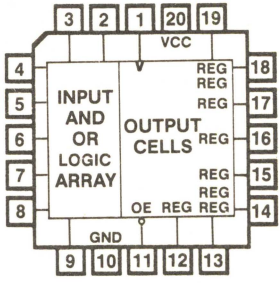



## PLCC Pinouts

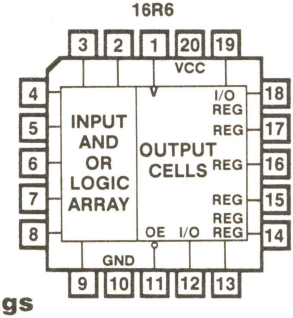
16L8



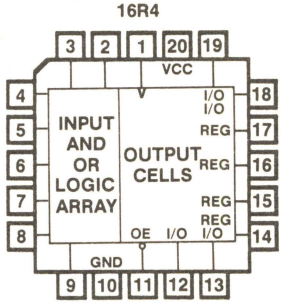

16R8




16R6




16R4




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 16R8, 16R6, 16R4 | 8 | 6 | | ns |
| | | High | | 8 | 5 | | |
| $t_{su}$ | Set up time from input or feedback to clock | | | 10 | 8 | | ns |
| $t_h$ | Hold time | | | 0 | −6 | | ns |
| $T_A$ | Operating free-air temperature | | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 120 | 180 | mA |
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 2.0 V at f = 1 MHz | | | 2 | | pF |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | 4 | | pF |
| $C_{CLK,EN}$ | Clock/enable capacitance | $V_{CLK,EN}$ = 2.0 V at f = 1 MHz | | | 9 | | pF |

1. The PAL16R8D Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8, 16R6, 16R4 | $R_1 = 200\,\Omega$<br>$R_2 = 390\,\Omega$ | 3 | 8 | 10 | ns |
| $t_{CLK}$ | Clock to output or feedback except 16L8 | | | | 2 | 6 | 7 | ns |
| $t_{CF}$ | Clock to feedback | | | | 2 | 5 | 6.5 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8 | | | | 3 | 8 | 10 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8 | | | | 3 | 8 | 10 | ns |
| $t_{EA}$ | Input to output enable | | 16L8, 16R6, 16R4 | | 1 | 8 | 10 | ns |
| $t_{ER}$ | Input to output disable | | 16L8, 16R6, 16R4 | | 1 | 8 | 10 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8, 16R6, 16R4 | | 58.8 | 71 | | MHz |
| | | Internal | | | 60 | 76 | | |
| | | No feedback | | | 62.5 | 90 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 10 | 6 | | ns |
| | | High | 10 | 5 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8B, 16R6B, 16R4B | 15 | 10 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 100 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 120 | 180 | mA |

1. The PAL16R8B Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8B, 16R6B, 16R4B | Commercial $R_1 = 200\ \Omega$ $R_2 = 390\ \Omega$ | | 12 | 15 | ns |
| $t_{CLK}$ | Clock to output or feedback except 16L8B | | | | | 8 | 12 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8B | | | | | 10 | 15 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8B | | | | | 10 | 15 | ns |
| $t_{EA}$ | Input to output enable | | 16L8B, 16R6B, 16R4B | | | 12 | 22 | ns |
| $t_{ER}$ | Input to output disable | | 16L8B, 16R6B, 16R4B | | | 12 | 15 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8B, 16R6B, 16R4B | | 37 | 45 | | MHz |
| | | No feedback | | | 50 | 55 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 5.25 V |
| Input voltage | -3.0 V to 7.0 V | -1.0 V to 14.0 V |
| Off-state output voltage | -0.5 V to 7.0 V | -0.5 V to 7.0 V |
| Output current into outputs | 8 mA | |
| Storage temperature | -65°C to +150°C | |
| Ambient temperature with power applied | -55°C to +125°C | |
| UV light exposure | 7258 W-sec/cm$^2$ | |
| Static discharge voltage | >2001 V | |
| Latchup current ($T_A$ = 0°C to 75°C) | >100 mA | |

## Operating Conditions

| SYMBOL | PARAMETER | | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | | 4.5 | 5 | 5.5 | V |
| $t_{wl}$ | Width of clock | Low | 16R8, 16R6, 16R4 | 15 | 10 | | ns |
| $t_{wh}$ | | High | | 15 | 10 | | ns |
| $t_{su}$ | Setup time from input or feedback to clock | | | 20 | 15 | | ns |
| $t_h$ | Hold time | | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2.0 | | | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | | -10 | μA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 10 | μA |
| $I_I$[3] | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 10 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.35 | 0.4 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = MAX | $V_O$ = 0 V | | | -300 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX, $V_I$ = GND. Outputs open | | | 30 | 45 | mA |
| $C_{IN}$ | Input capacitance[5] | $V_{IN}$ =0 V at f = 1 MHz | | | 5 | 7 | pF |
| $C_{OUT}$ | Output capacitance[5] | $V_{OUT}$ =0 V at f = 1 MHz | | | 5 | 7 | pF |

1. The PALC16R8Q-25 Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$). For pin 1 $I_{IH}$ = 25 μA max, $I_I$ = 1 mA max.
4. No more than one output should be shorted at a time, and duration of the short-circuit should not exceed one second.
5. Sampled but not 100% tested.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | 16L8, 16R6, 16R4 | | $R_1$ = 560 Ω, $R_2$ = 1.1K Ω | | 20 | 25 | ns |
| $t_{CLK}$ | CLK to output | | 16R8, 16R6, 16R4 | | | 10 | 15 | ns |
| $t_{CF}$ | CLK to feedback | | | | | 9 | 13 | ns |
| $t_{PZX}$ | Pin 11 to output enable | | | | | 15 | 20 | ns |
| $t_{PXZ}$ | Pin 11 to output disable | | | | | 15 | 20 | ns |
| $t_{EA}$ | Input to output enable | 16L8, 16R6, 16R4 | | | | 20 | 25 | ns |
| $t_{ER}$ | Input to output disable | | | | | 20 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External feedback ($1/t_{su} + t_{CLK}$) | 16R8, 16R6, 16R4 | | 28.5 | 40 | | MHz |
| | | Internal feedback ($1/t_{su} + t_{CF}$) | | | 30 | 40 | | |
| | | No feedback ($1/t_{wh} + t_{wl}$) | | | 33.3 | 50 | | |

### Output Register Preload

The preload function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct loading of output states. The procedure is:

1. Raise VCC to 5.0 V ± 0.5 V.
2. Disable output registers by setting pin 11 to VIH .
3. Apply VIL/VIH as desired to all registered output pins. Leave combinatorial outputs floating.
4. Pulse pin 5 from VIL to 13.5 V to VIL.
5. Remove VIL/VIH from all registered output pins.
6. Enable output registers.
7. Verify for VOL/VOH at all registered output pins.

### Programming and Erasing

The PALC16R8Q-25 Series can be programmed on standard logic programmers. The PALC16R8Q-25 Series may be erased by ultraviolet light when contained in the windowed package.

For erasure, the recommended ultraviolet light wavelength is 2537 Angstroms. The minimum dose required is 25,000 mW-sec/cm$^2$ (UV intensity x exposure time). For an ultraviolet lamp with a 12 mW/cm$^2$ power rating, the minimum exposure time would be 25,000/12 seconds = 35 minutes. The device needs to be within one inch of the lamp during erasure.

Permanent damage may result if the device is exposed to high-intensity UV light for an extended period of time. The recommended maximum dosage is 7258 W-sec/cm$^2$.

Wavelengths of light less than 4000 Angstroms can partially erase the device in the windowed package. For this reason, an opaque label should be placed over the window, especially if the device will be exposed to sunlight or fluorescent lighting for extended periods of time.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 15 | 10 | | ns |
| | | High | 15 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8B-2, 16R6B-2, 16R4B-2 | 25 | 15 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −100 | −250 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 60 | 90 | mA |

1. The PAL16R8B-2 Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8B-2, 16R6B-2, 16R4B-2 | Commercial $R_1 = 200\ \Omega$ $R_2 = 390\ \Omega$ | | 17 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback except 16L8B-2 | | | | | 10 | 15 | ns |
| $t_{CF}$ | Clock to feedback except 16L8B-2 | | | | | 8 | 10 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8B-2 | | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8B-2 | | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 16L8B-2, 16R6B-2, 16R4B-2 | | | 10 | 25 | ns |
| $t_{ER}$ | Input to output disable | | 16L8B-2, 16R6B-2, 16R4B-2 | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8B-2, 16R6B-2, 16R4B-2 | | 25 | 40 | | MHz |
| | | Internal | | | 28.5 | 43 | | |
| | | No feedback | | | 33 | 50 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 15 | 10 | | ns |
| | | High | 15 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8, 16R6, 16R4 | 25 | 15 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 120 | 180 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short curcuit should not exceed one second.

5

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16R6A, 16R4A, 16L8A | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 15 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 10 | 15 | ns |
| $t_{CF}$ | Clock to feedback | | | | | 8 | 10 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8A | | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8A | | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 16R6A, 16R4A, 16L8A | | | 10 | 25 | ns |
| $t_{ER}$ | Input to output disable | | 16R6A, 16R4A, 16L8A | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8A, 16R6A, 16R4A | | 25 | 40 | | MHz |
| | | Internal | | | 28.5 | 40 | | |
| | | No feedback | | | 33 | 50 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 25 | 10 | | ns |
| | | High | 25 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8B-4, 16R6B-4, 16R4B-4 | 35 | 25 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −1 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −100 | −250 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 30 | 55 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

5

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8B-4, 16R6B-4, 16R4B-4 | $R_1$ = 800 Ω<br>$R_2$ = 1.56 KΩ | | 25 | 35 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 15 | 25 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8B-4 | | | | | 15 | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8B-4 | | | | | 15 | 25 | ns |
| $t_{EA}$ | Input to output enable | | 16L8B-4, 16R6B-4, 16R4B-4 | | | 25 | 35 | ns |
| $t_{ER}$ | Input to output disable | | 16L8B-4, 16R6B-4, 16R4B-4 | | | 25 | 35 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8B-4, 16R6B-4, 16R4B-4 | | 16 | 25 | | MHz |
| | | No feedback | | | 25 | 50 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 25 | 10 | | ns |
| | | High | 25 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8A-2, 16R6A-2, 16R4A-2 | 35 | 25 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −1 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −100 | −250 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 60 | 90 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

5

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8A-2, 16R6A-2, 16R4A-2 | $R_1 = 200\ \Omega$<br>$R_2 = 390\ \Omega$ | | 25 | 35 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 15 | 25 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8A-2 | | | | | 15 | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8A-2 | | | | | 15 | 25 | ns |
| $t_{EA}$ | Input to output enable | | 16L8A-2, 16R6A-2, 16R4A-2 | | | 25 | 35 | ns |
| $t_{ER}$ | Input to output disable | | 16L8A-2, 16R6A-2, 16R4A-2 | | | 25 | 35 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8A-2, 16R6A-2, 16R4A-2 | | 16 | 25 | | MHz |
| | | No feedback | | | 20 | 50 | | |

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 30 | 20 | | ns |
| | | High | 30 | 20 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 16R8A-4, 16R6A-4, 16R4A-4 | 60 | 45 | | ns |
| $t_h$ | Hold time | | 0 | −15 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −1 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 30 | 50 | mA |

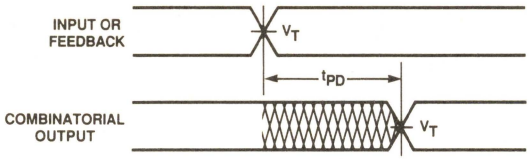
1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

5

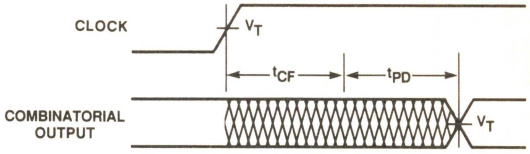
## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 16L8A-4, 16R6A-4, 16R4A-4 | $R_1$ = 800 Ω<br>$R_2$ = 1.56 KΩ | | 35 | 55 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 20 | 35 | ns |
| $t_{PZX}$ | Pin 11 to output enable except 16L8A-4 | | | | | 15 | 30 | ns |
| $t_{PXZ}$ | Pin 11 to output disable except 16L8A-4 | | | | | 15 | 30 | ns |
| $t_{EA}$ | Input to output enable | | 16L8A-4, 16R6A-4, 16R4A-4 | | | 30 | 50 | ns |
| $t_{ER}$ | Input to output disable | | 16L8A-4, 16R6A-4, 16R4A-4 | | | 30 | 50 | ns |
| $f_{MAX}$ | Maximum frequency | External | 16R8A-4, 16R6A-4, 16R4A-4 | | 11 | 18 | | MHz |
| | | No feedback | | | 16 | 25 | | |

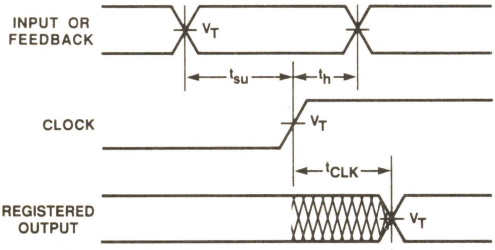
## Switching Waveforms



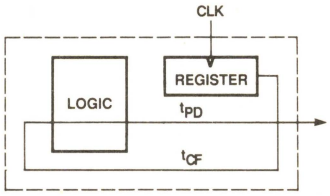

Combinatorial Output



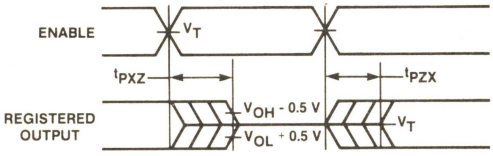

Clock to Feedback to Combinatorial Output (See Path Below)



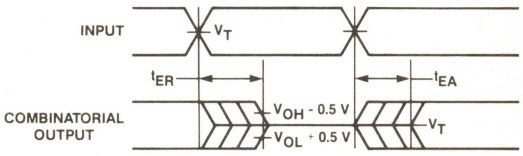

Registered Output



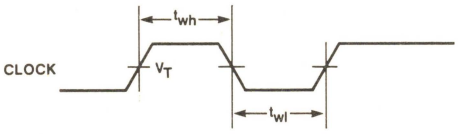

Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical




Pin 11 to Output Disable/Enable




Input to Output Disable/Enable




Clock Width

## Key to Timing Diagrams

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE; CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

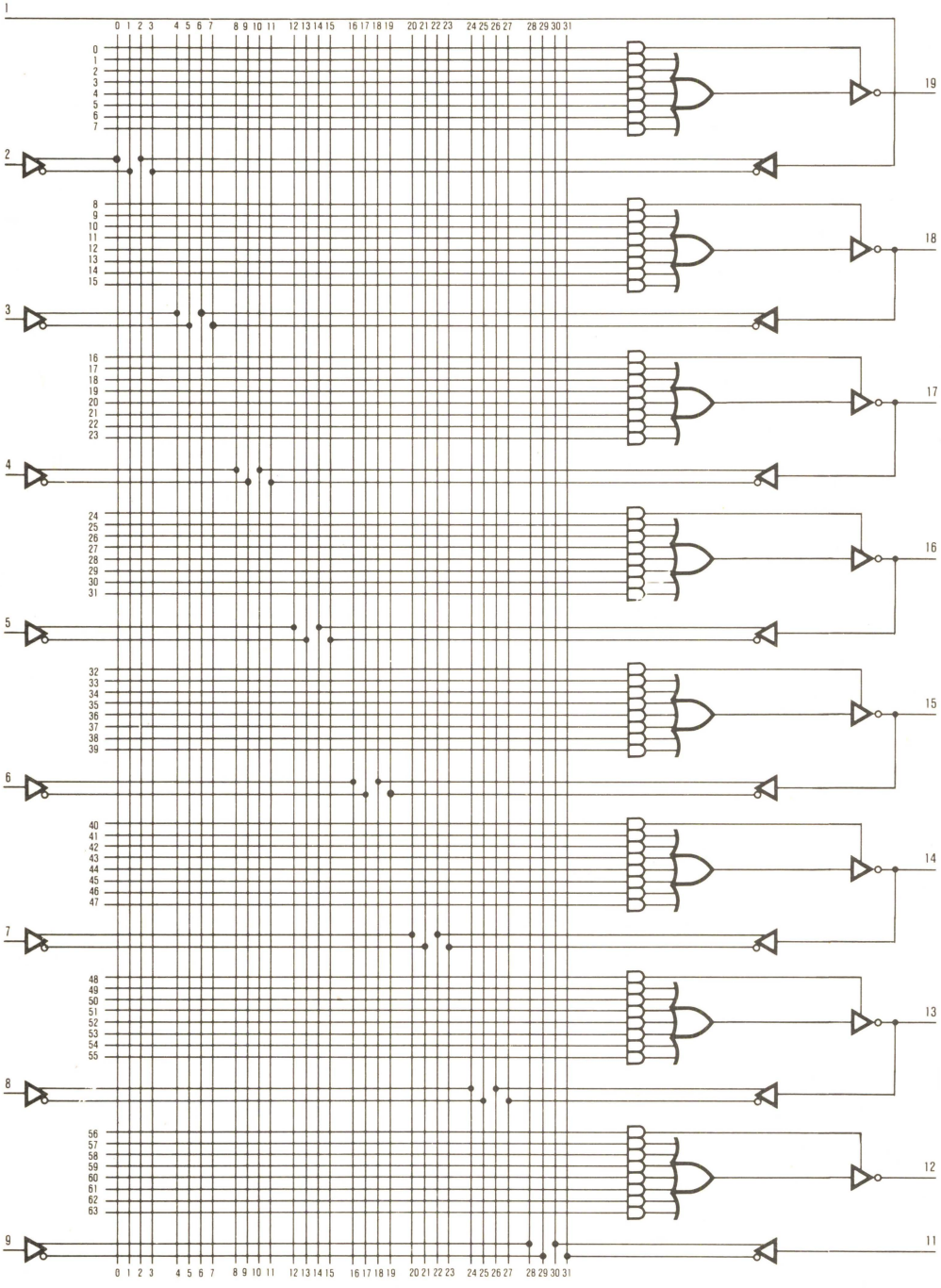
### Switching Test Load
(refer to page 5-164)

### Programmers/Development Systems
(refer to Programmer Reference Guide, page 3-81)

### Schematic of Inputs and Outputs
(refer to page 5-164)

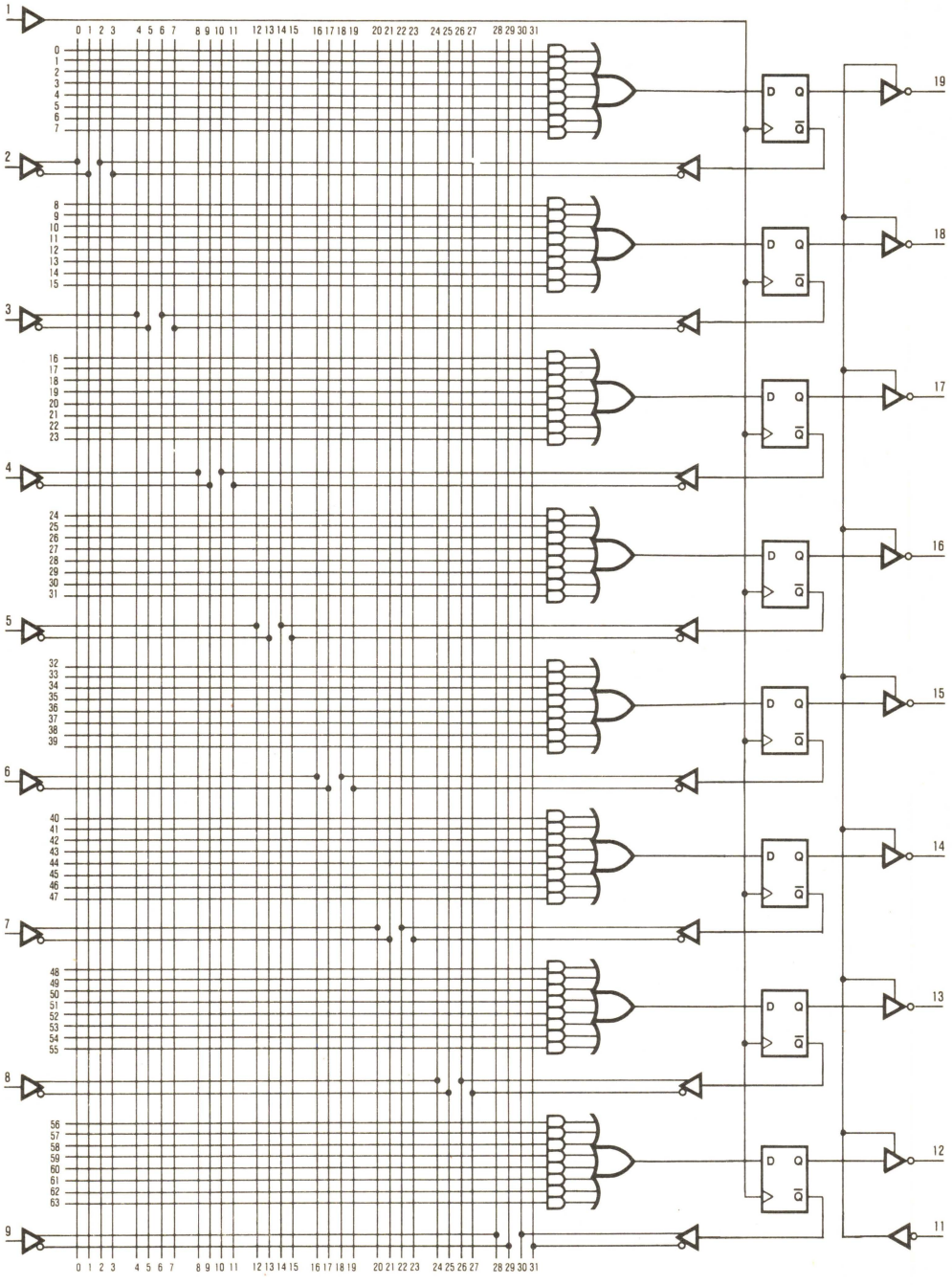
## Logic Diagram

16L8



## Logic Diagram

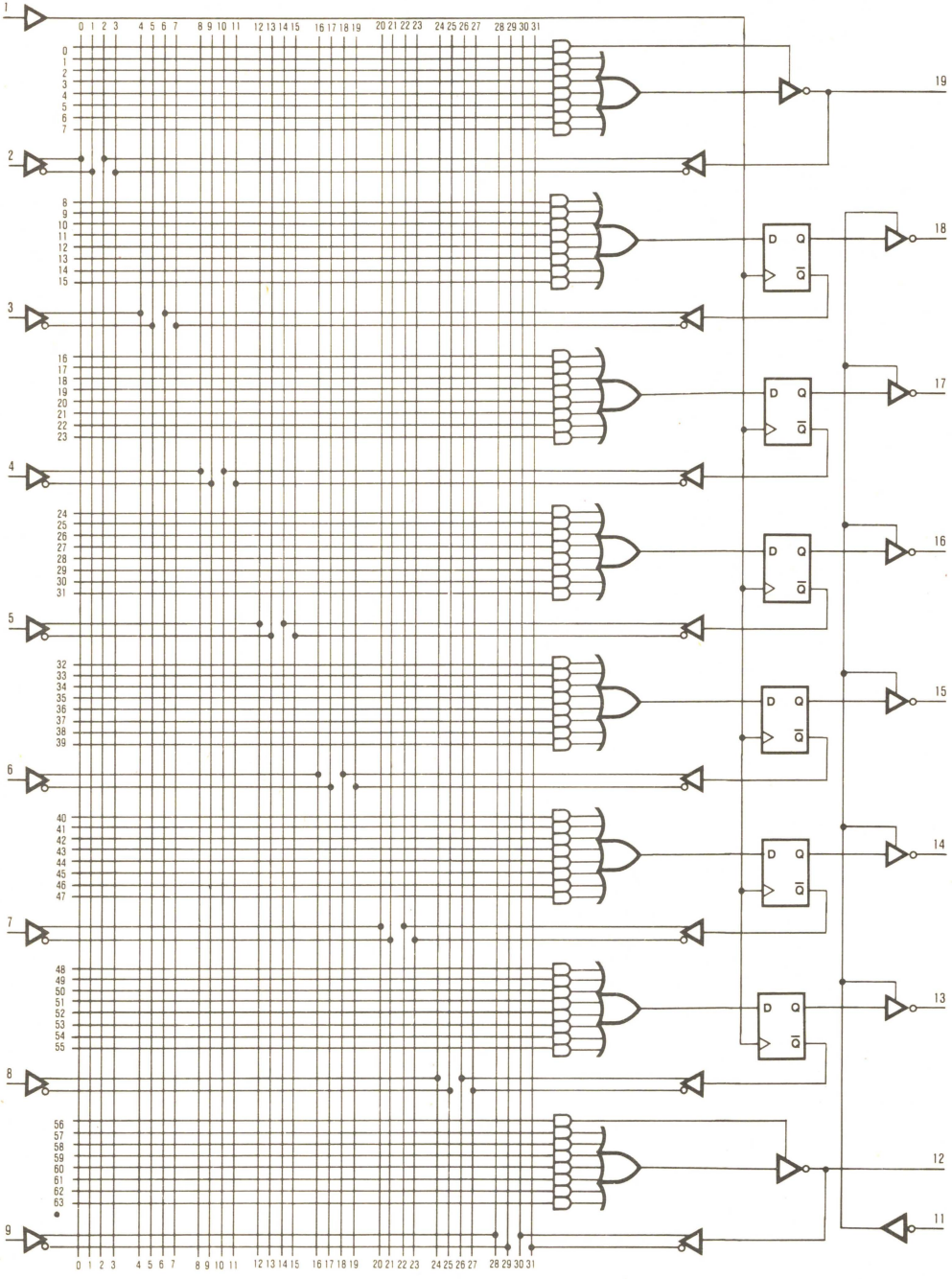
16R8




5

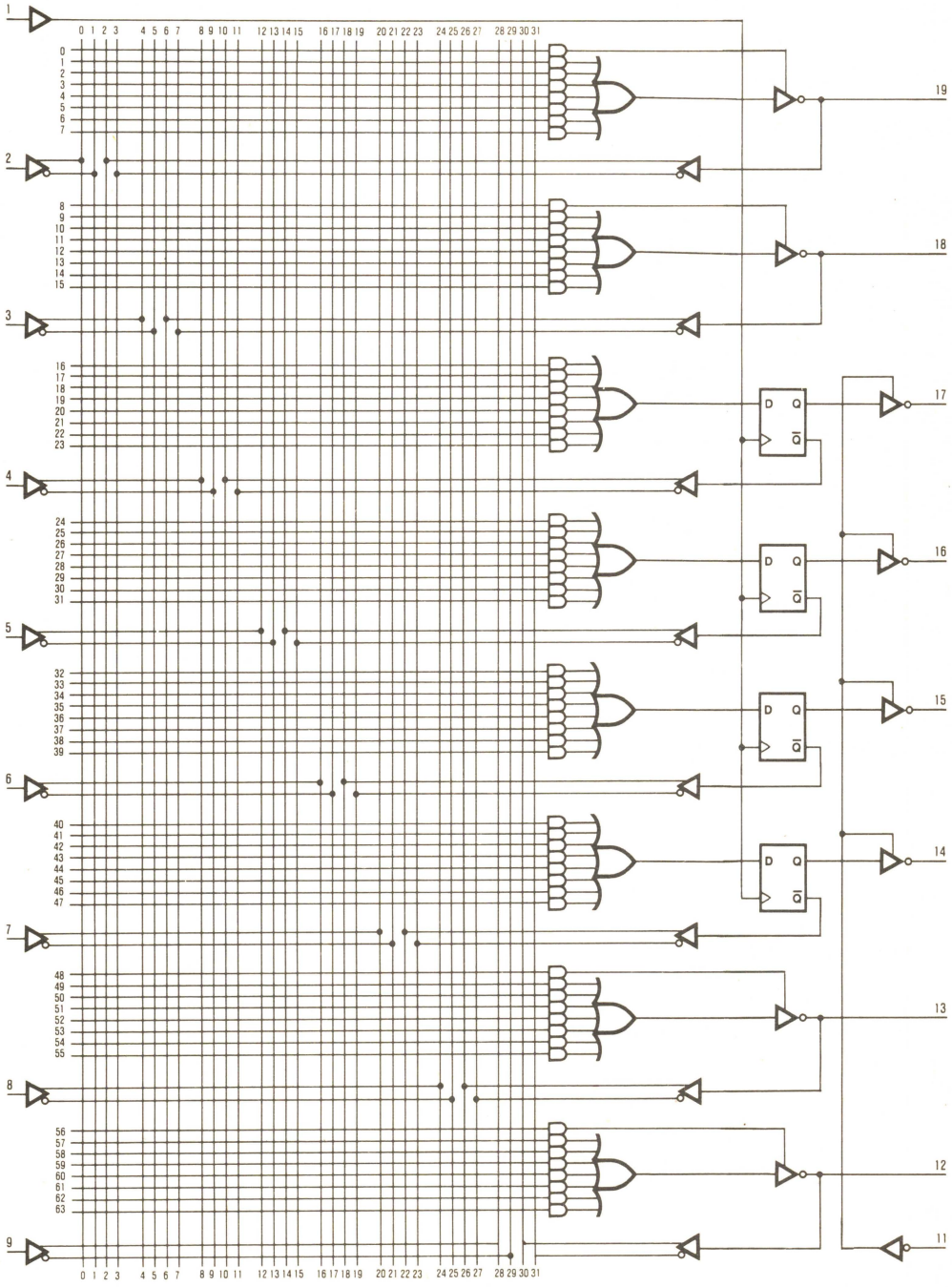
## Logic Diagram

16R6



## Logic Diagram

16R4




5

# Zero Standby Power CMOS ZPAL™ Family

# PALC16R8Z-25 Series

## ADVANCE INFORMATION

### Features/Benefits

- CMOS technology provides zero standby power: 100 µA max
- Low CMOS operating power
- 28.5 MHz max frequency and 25-ns propagation delay match bipolar speeds
- UV-erasable cell technology provides highest programming and functional yields
- Register preload eases device testing
- Programmable replacement for CMOS/TTL logic
- Instant prototyping and easier board layout
- Reduces chip count by greater than four to one
- Erasable windowed 20-pin package
- Cost effective OTP Plastic DIP package and 20-pin Plastic Leaded Chip Carrier
- Programmed on standard PAL® device programmers
- Security bit prevents duplication by competitors

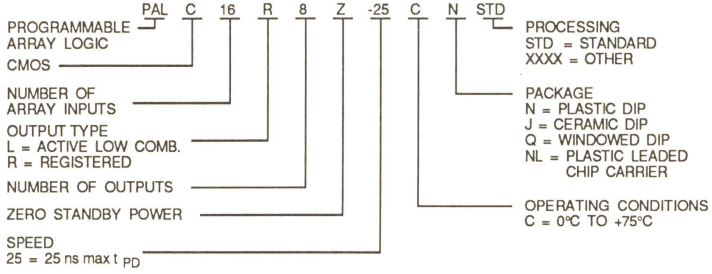
### Logic Array Description

| PART NUMBER | PKG | LOGIC ARRAY | | |
|---|---|---|---|---|
| | | ARRAY INPUTS | OUTPUTS | |
| | | | COMB | REG |
| PALC16L8Z | Q, N J, NL | 16 | 8 | – |
| PALC16R8Z | | 16 | – | 8 |
| PALC16R6Z | | 16 | 2 | 6 |
| PALC16R4Z | | 16 | 4 | 4 |

### Description

The CMOS PALC16R8Z Series is a CMOS version of the bipolar 20-pin PAL16R8 Series. Monolithic Memories' advanced CMOS technology provides high speed and very low power. Standby power consumption is typically less than 10 µA. Active power rises at less than 3 mA per MHz of operating frequency. The PAL16R8 Series is completely pin compatible with the 16L8, 16R8, 16R6, and 16R4 architectures. The PAL16R8Z Series can thus be used in existing 20-pin PAL device sockets.

### Ordering Information




631 01

# Arithmetic Series
# PAL16X4

## Features/Benefits

- Bit-pair decoding
- Easy generation of arithmetic operations
- Security fuse

## Description

The PAL16X4 has arithmetic gated feedback. This is a specialized device for arithmetic applications.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

The PAL16X4 has eight dedicated input lines, and each combinatorial output is an I/O pin. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Ordering Information

PAL16X4 C N STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- OUTPUTS TYPE
  - X = XOR-gated
  - Arithmetic
  - Registered Feedback
- NUMBER OF OUTPUTS
- PROCESSING
  - STD = Standard
  - XXXX = Other
- PACKAGE
  - N = Plastic DIP
  - J = Ceramic DIP
- OPERATING CONDITIONS
  - C = Commercial

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Arithmetic Gated Feedback

The arithmetic functions (add, subtract, greater than, and less than) are implemented by addition of gated feedback to the features of the XOR PAL device. The XOR at the input of the D-type flip-flop allows carrys from previous operations to be XORed with two variable sums generated by the PAL device array. The flip-flop Q output is fed back to be gated with input terms A (Figure 1). This gated feedback provides any one of the sixteen possible Boolean combinations which are mapped in the Karnaugh map (Figure 2). Figure 3 shows how the PAL device array can be programmed to perform these sixteen operations. These features provide for versatile operations on two variables and facilitate the parallel generation of carrys necessary for fast arithmetic operations.

5

## PAL Arithmetic Series

| | ARRAY INPUTS | OUTPUTS | | PRODUCT TERMS |
|---|---|---|---|---|
| | | COMBINATORIAL | REGISTERED | |
| PAL16X4 | 16 | 4 | 4 | 64 |

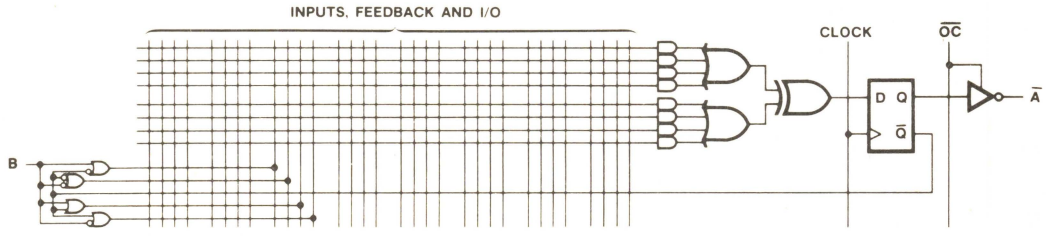



Figure 1.

10280A
JANUARY 1988

| $(\bar{A}+B)\cdot(\bar{A}+\bar{B})\cdot(A+\bar{B})\cdot(A+B)$ | -- | -x | xx | x- |
|---|---|---|---|---|
| -- | 1 | $\bar{A}+\bar{B}$ | $\bar{A}$ | $\bar{A}+B$ |
| -x | A+B | A:+:B | $\bar{A}\cdot B$ | B |
| xx | A | $A\cdot\bar{B}$ | 0 | $A\cdot B$ |
| x- | $A+\bar{B}$ | $\bar{B}$ | $\bar{A}\cdot\bar{B}$ | A:•:B |

Figure 2.

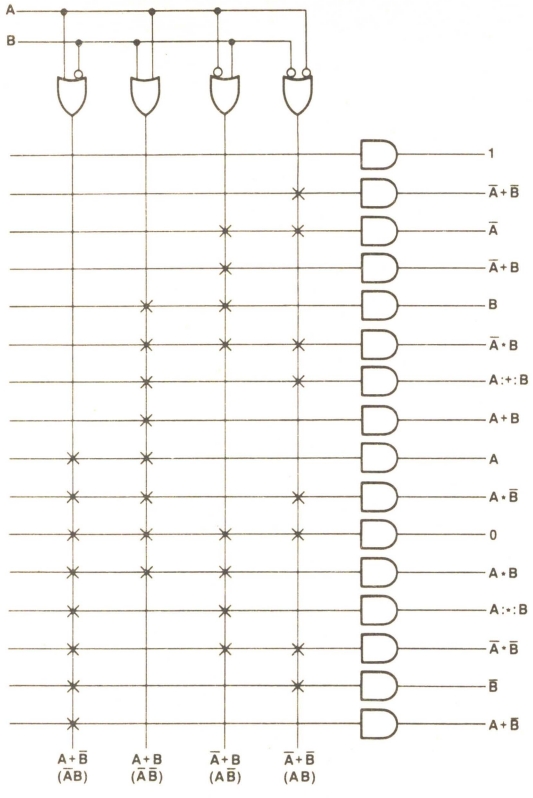


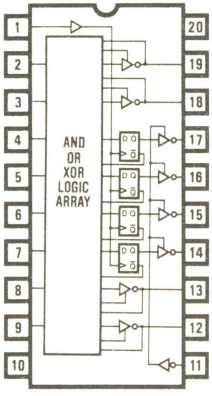

Figure 3.

## Packages

The commercial PAL16X4 is available in the DIP (N) and ceramic DIP (J) packages.

## DIP Pinout

16X4




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 25 | 10 | | ns |
| | | High | 25 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | | 45 | 30 | | ns |
| $t_h$ | Hold time | | 0 | -15 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

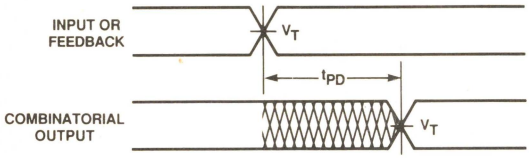
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 160 | 225 | mA |

## Switching Characteristics Over Operating Conditions

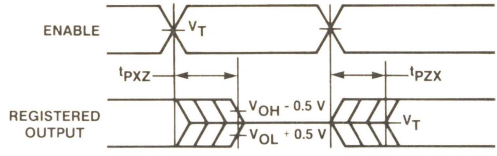
| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | $R_1$ = 200 Ω, $R_2$ = 390 Ω | | 30 | 40 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | 15 | 25 | ns |
| $t_{PZX}$ | Pin 11 to output enable | | | | 15 | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable | | | | 15 | 25 | ns |
| $t_{EA}$ | Input to output enable | | | | 30 | 40 | ns |
| $t_{ER}$ | Input to output disable | | | | 30 | 40 | ns |
| $f_{MAX}$ | Maximum frequency | External | | 14 | 22 | | MHz |
| | | No feedback | | 20 | 50 | | |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

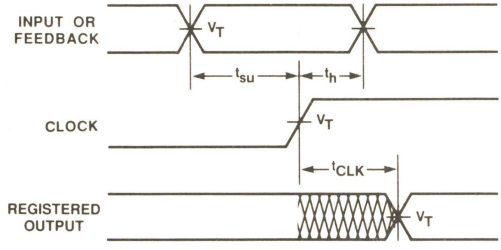
## Switching Waveforms



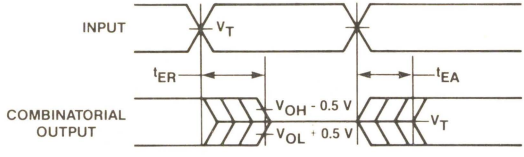

Combinatorial Output



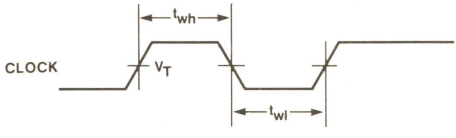

Pin 11 to Output Disable/Enable




Registered Output




Input to Output Disable/Enable




Clock Width

Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagrams

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE: CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

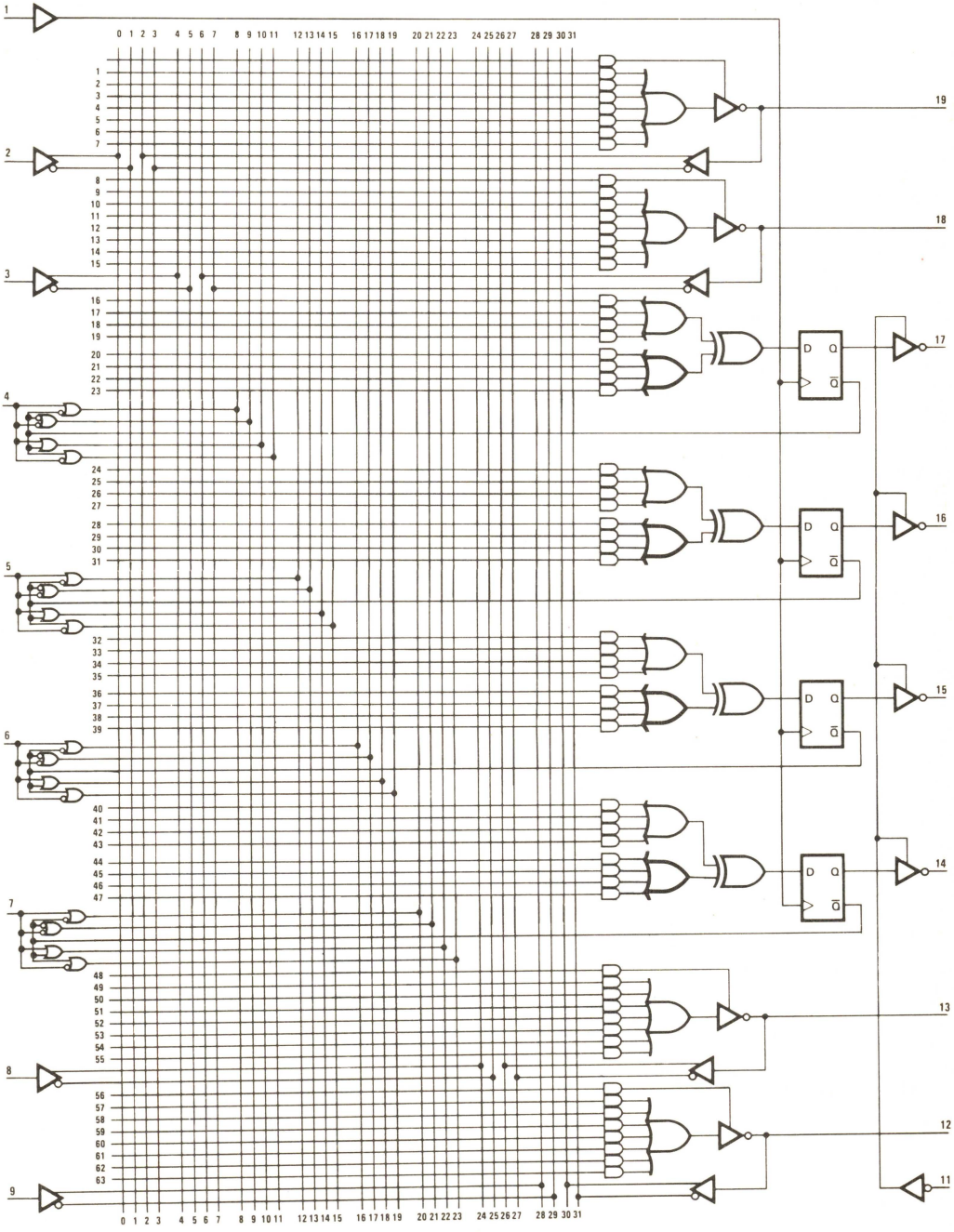
(refer to Programmer Reference Guide, page 3-81)

## Schematic of Inputs and Outputs

(refer to page 5-164)

## Logic Diagram

16X4



# Combinatorial PAL10H8 Series

## 10H8, 12H6, 14H4, 16H2
## 16C1
## 10L8, 12L6, 14L4, 16L2

### Features/Benefits

- Combinatorial architectures
- Active high or active low options
- Security fuse

### Ordering Information

PAL10H8 C N STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE
  - H = Active High Combinatorial
  - L = Active Low Combinatorial
  - C = Complementary Combinatorial
- NUMBER OF OUTPUTS
- OPERATING CONDITIONS
  - C = Commercial
- PACKAGE
  - N = Plastic DIP
  - J = Ceramic DIP
  - NL = Plastic Leaded Chip Carrier
  - SG = Small-Outline Gull-wing
- PROCESSING
  - STD = Standard
  - XXXX = Other

| | INPUTS | OUTPUTS | POLARITY | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|---|
| PAL10H8 | 10 | 8 | HIGH | 35 | 90 |
| PAL12H6 | 12 | 6 | HIGH | 35 | 90 |
| PAL14H4 | 14 | 4 | HIGH | 35 | 90 |
| PAL16H2 | 16 | 2 | HIGH | 35 | 90 |
| PAL16C1 | 16 | 2 | BOTH | 40 | 90 |
| PAL10L8 | 10 | 8 | LOW | 35 | 90 |
| PAL12L6 | 12 | 6 | LOW | 35 | 90 |
| PAL14L4 | 14 | 4 | LOW | 35 | 90 |
| PAL16L2 | 16 | 2 | LOW | 35 | 90 |

### Description

The PAL10H8 Series is made up of nine combinatorial 20-pin PAL devices. They implement simple combinatorial logic, with no feedback. Each has sixteen product terms total, divided among the outputs, with two to sixteen product terms per output.

### Polarity

Both active high and active low versions are available for each architecture. The 16C1 offers both polarities of its single output.

### Performance

The standard series has a propagation delay (tpd) of 35 nanoseconds (ns), except for the 16C1 at 40 ns. Standard supply current is 90 milliamps (mA).

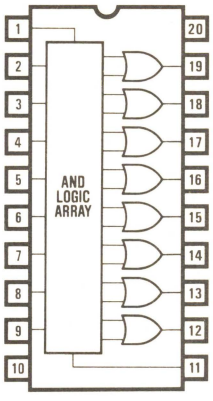
### Packages

The commercial PAL10H8 Series is available in the plastic DIP (N), ceramic DIP (J), plastic leaded chip carrier (NL), and small outline (SG) packages.

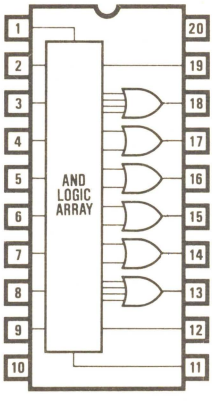
10285A
JANUARY 1988

## DIP/SO Pinouts

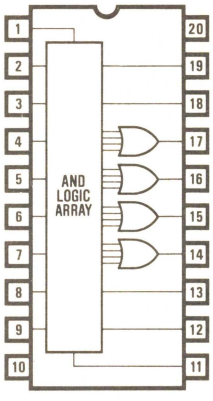
10H8



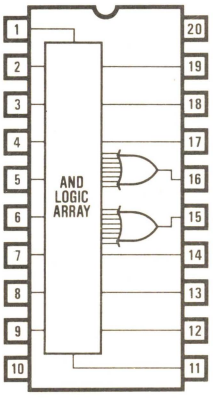

12H6



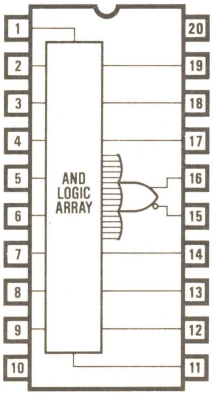

14H4



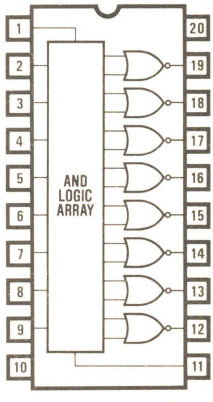

16H2



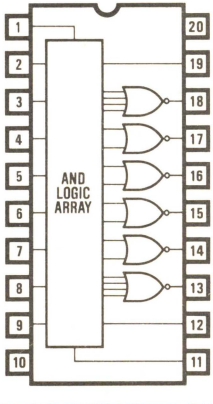

16C1



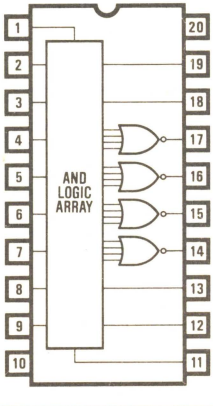

10L8



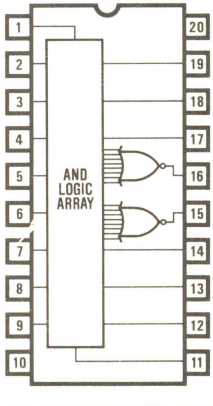

12L6




14L4



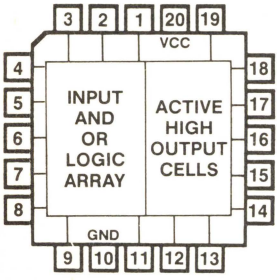

16L2



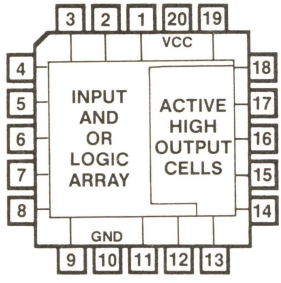

5

## PLCC Pinouts

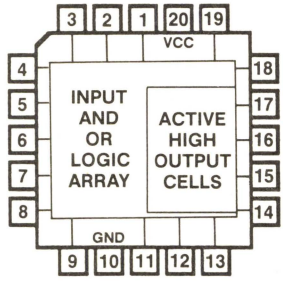
10H8



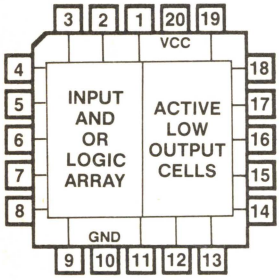

12H6



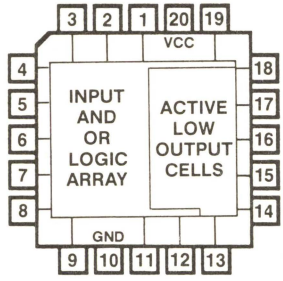

14H4



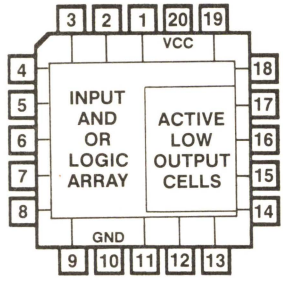

10L8



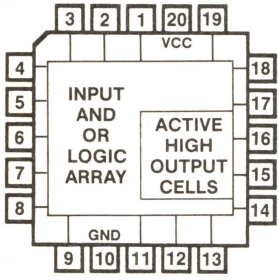

12L6



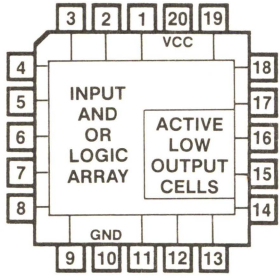

14L4



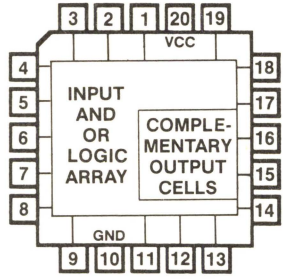

16H2




16L2




16C1




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.75 | 5 | 5.25 | V |
| $T_A$ | Operating free-air temperature | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

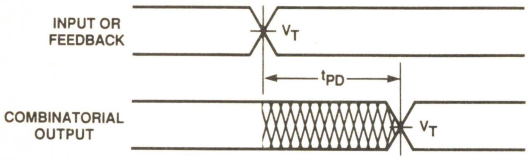
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OS}$[2] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 55 | 90 | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | Except 16C1 | $R_1$ = 560 Ω<br>$R_2$ = 1.1 kΩ | | 25 | 35 | ns |
| | | 16C1 | | | 25 | 40 | |

1. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Waveforms




Combinatorial Output

Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.

## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

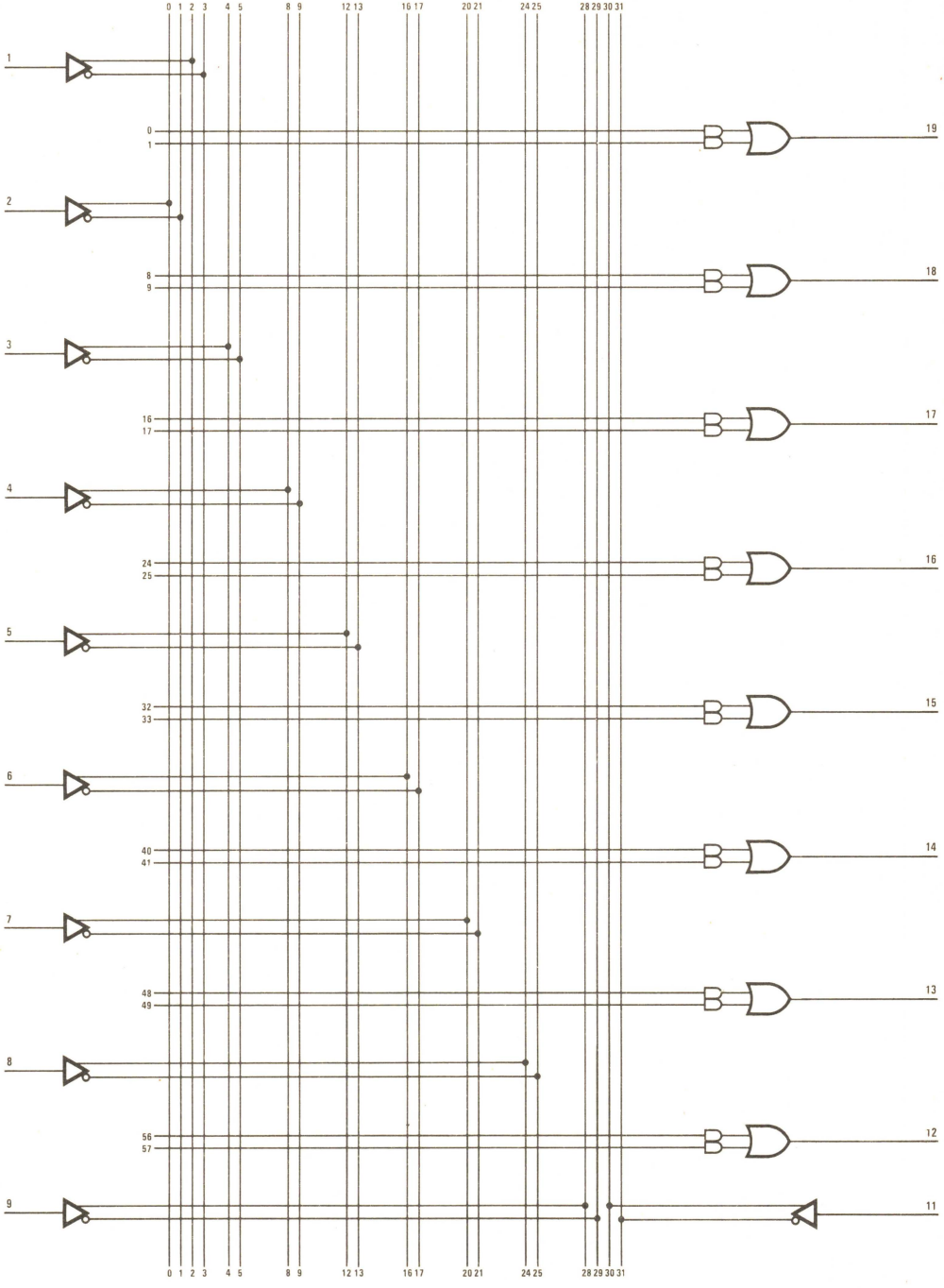
(refer to Programmer Reference Guide, page 3-81)

## Schematic of Inputs and Outputs

(refer to page 5-164)

## Logic Diagram

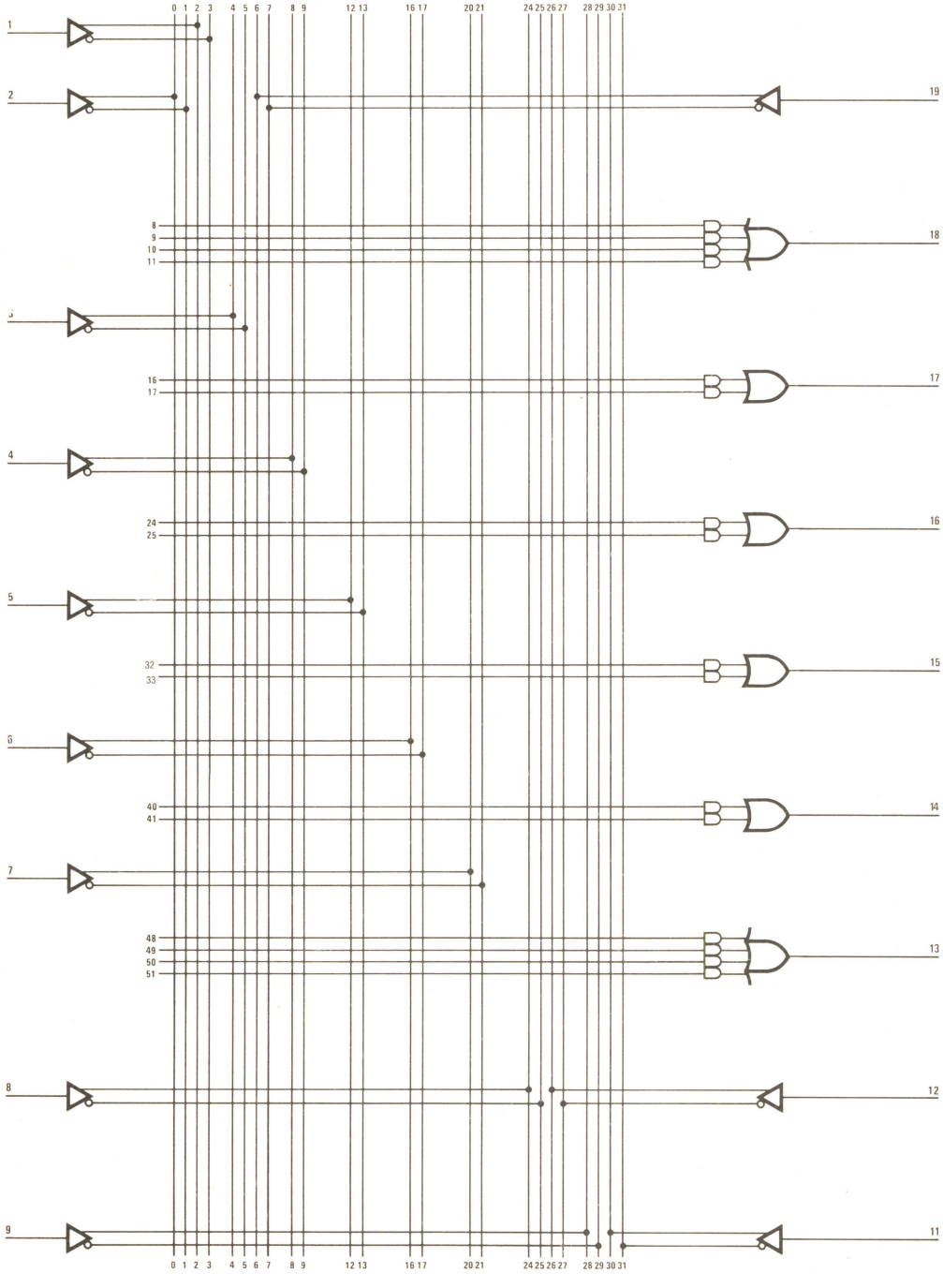
### 10H8




5

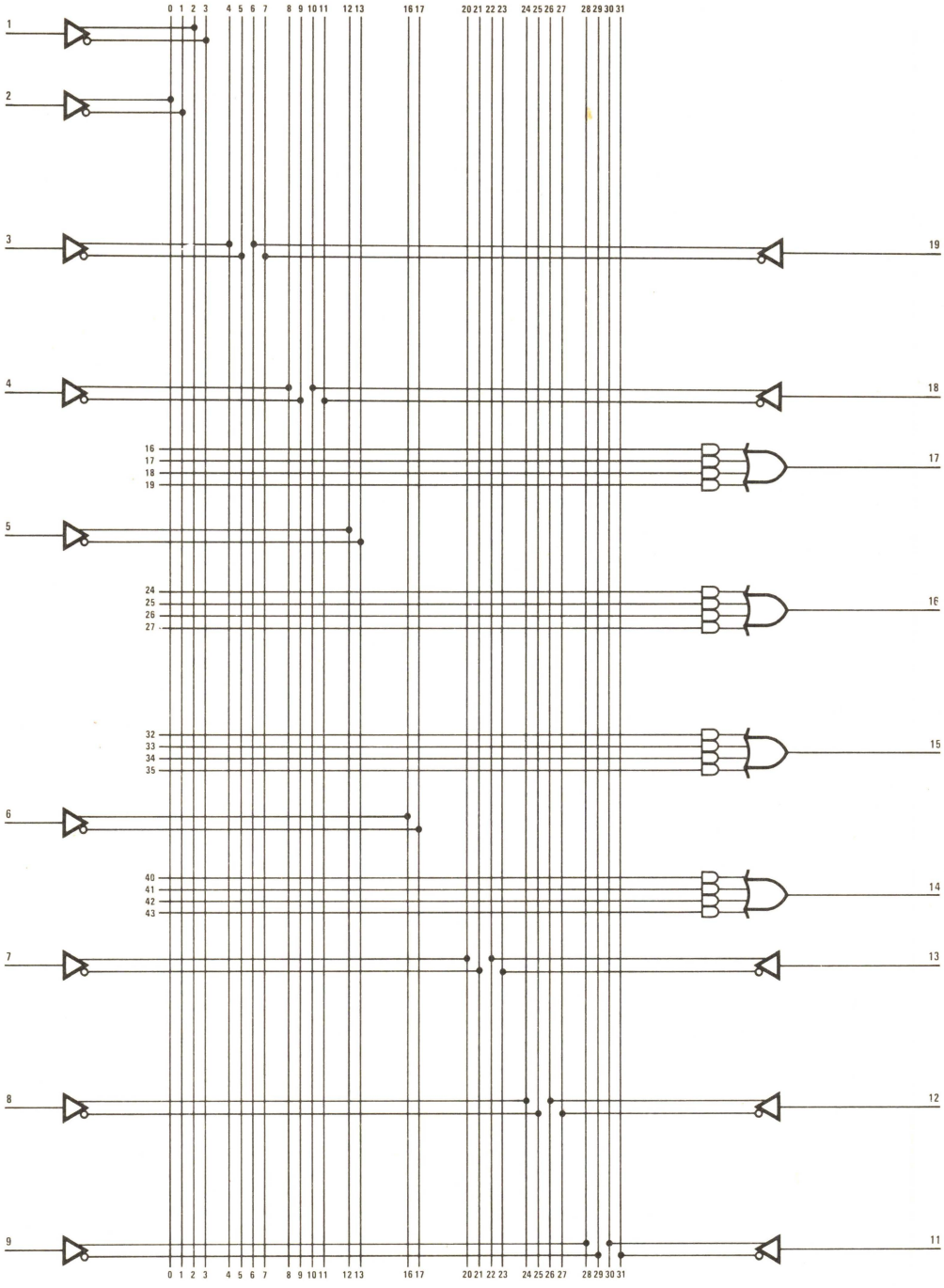
## Logic Diagram

### 12H6



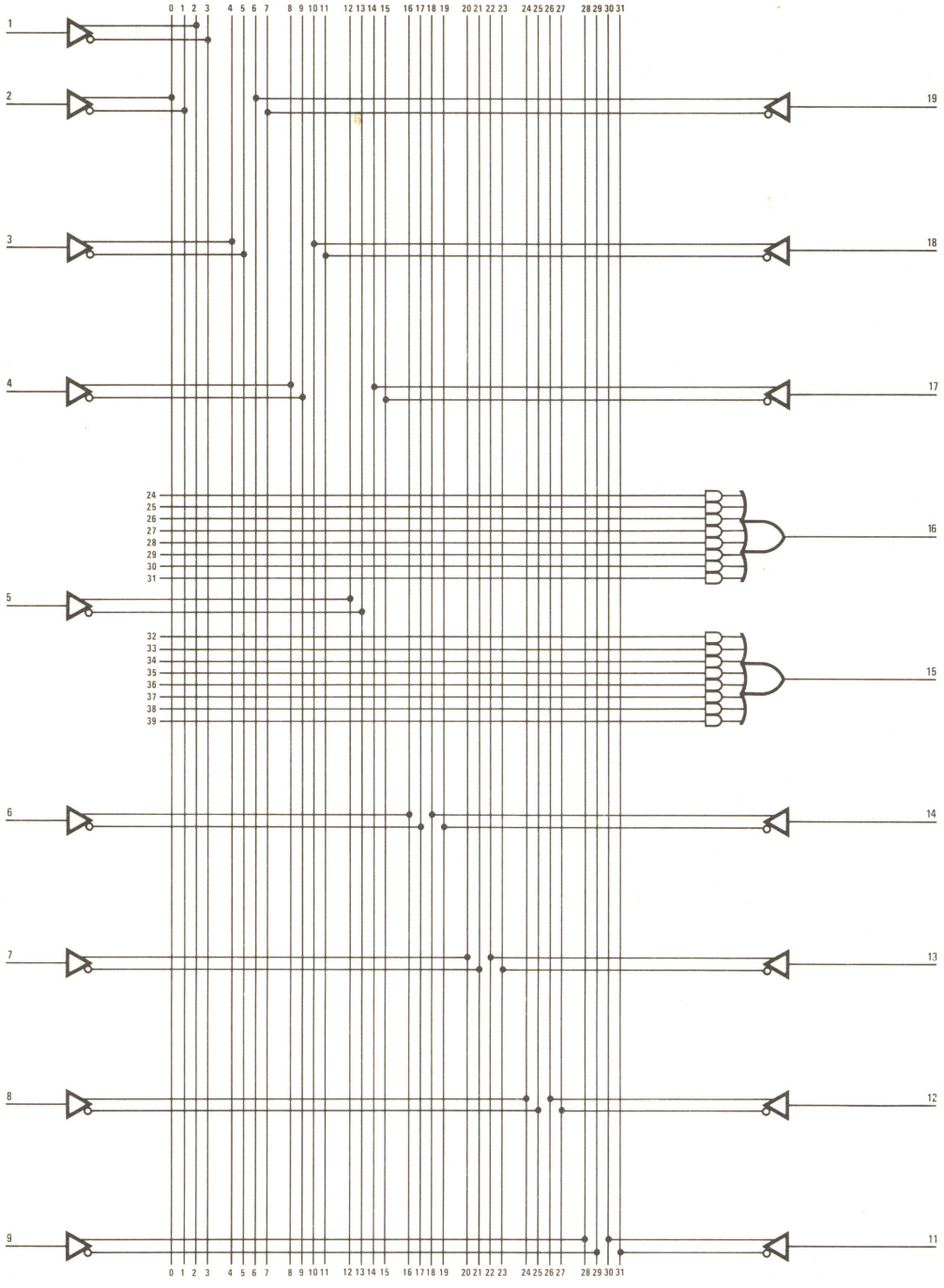
## Logic Diagram

### 14H4



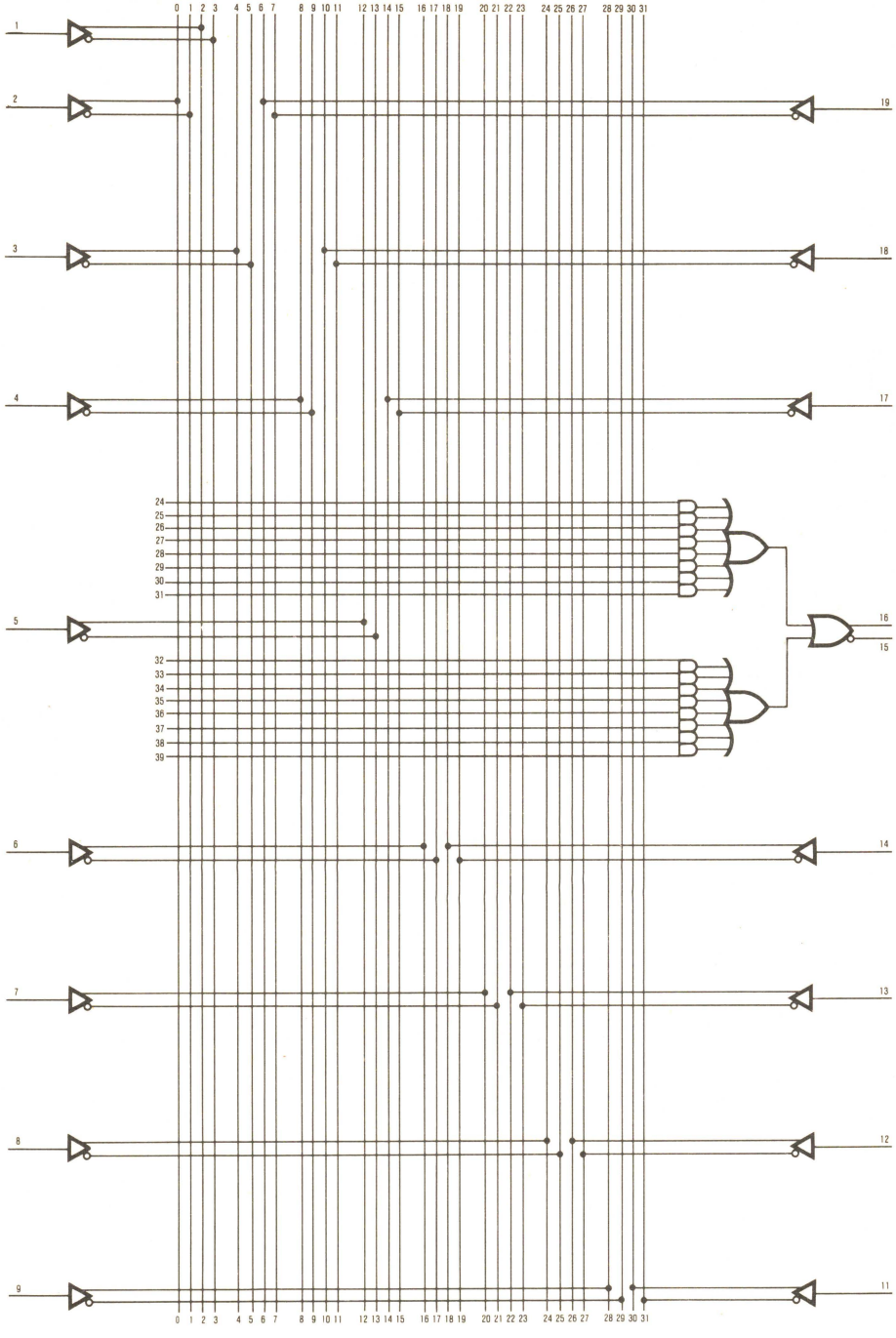
## Logic Diagram

### 16H2



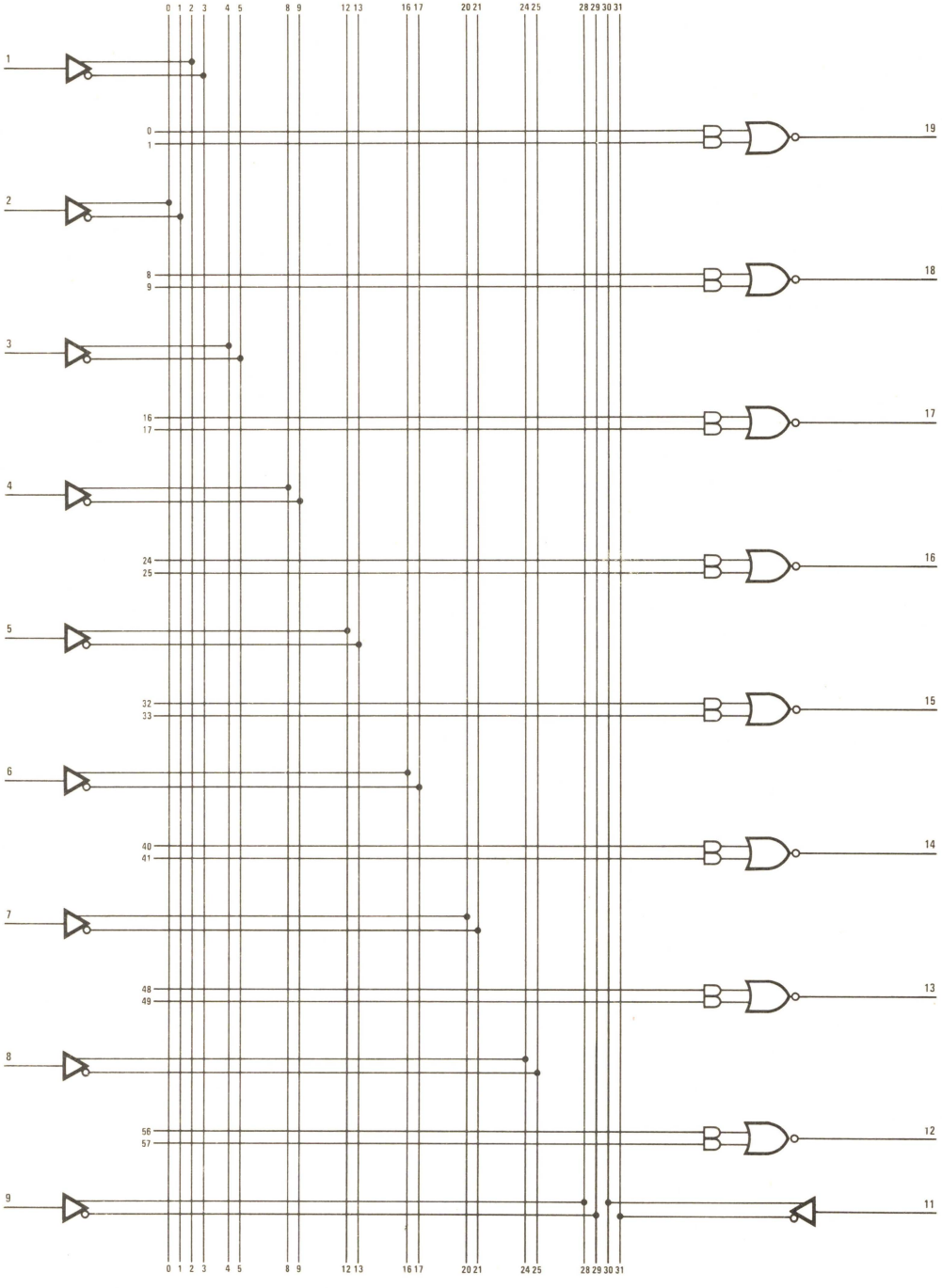
## Logic Diagram

### 16C1



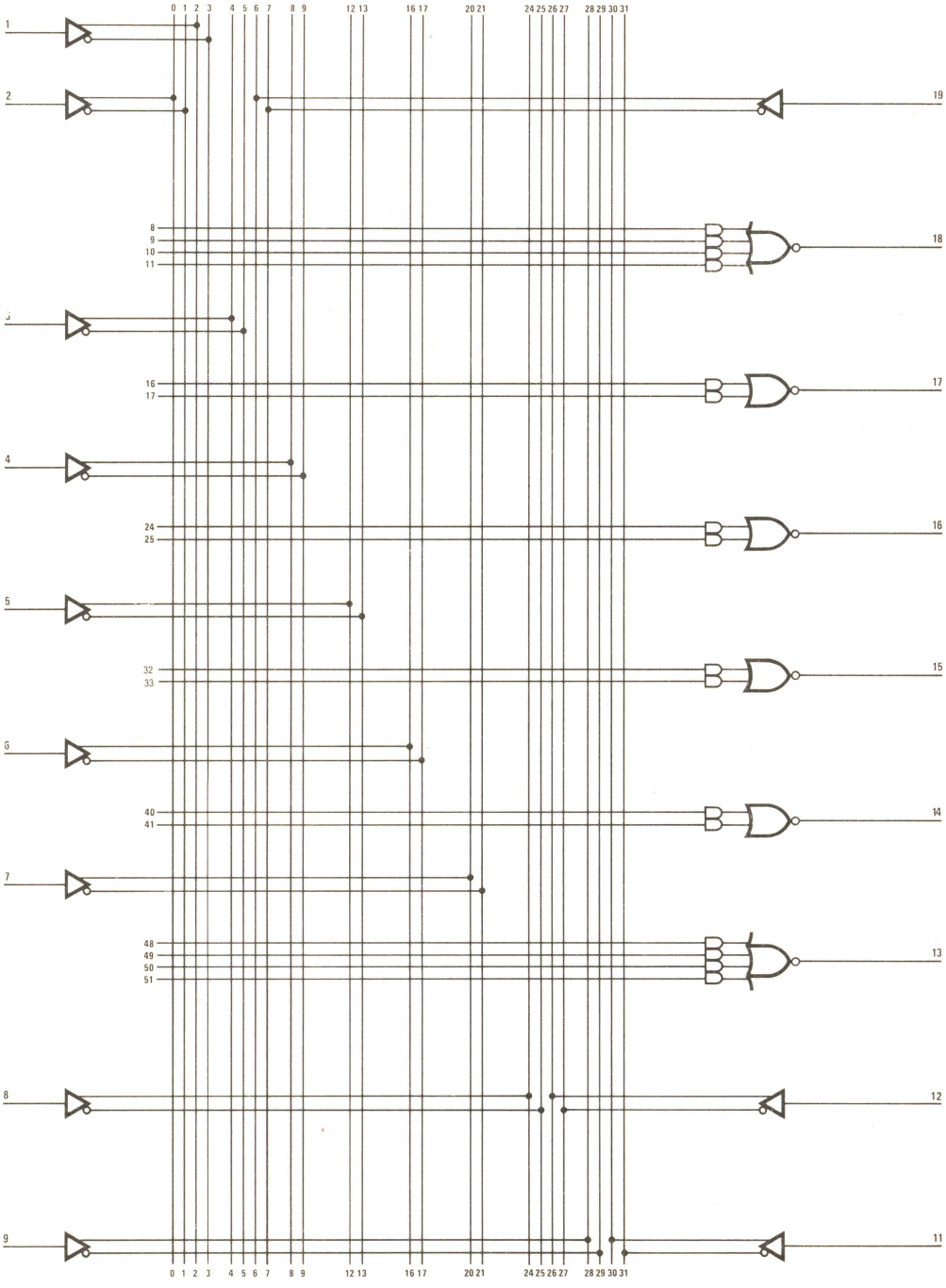
## Logic Diagram

### 10L8



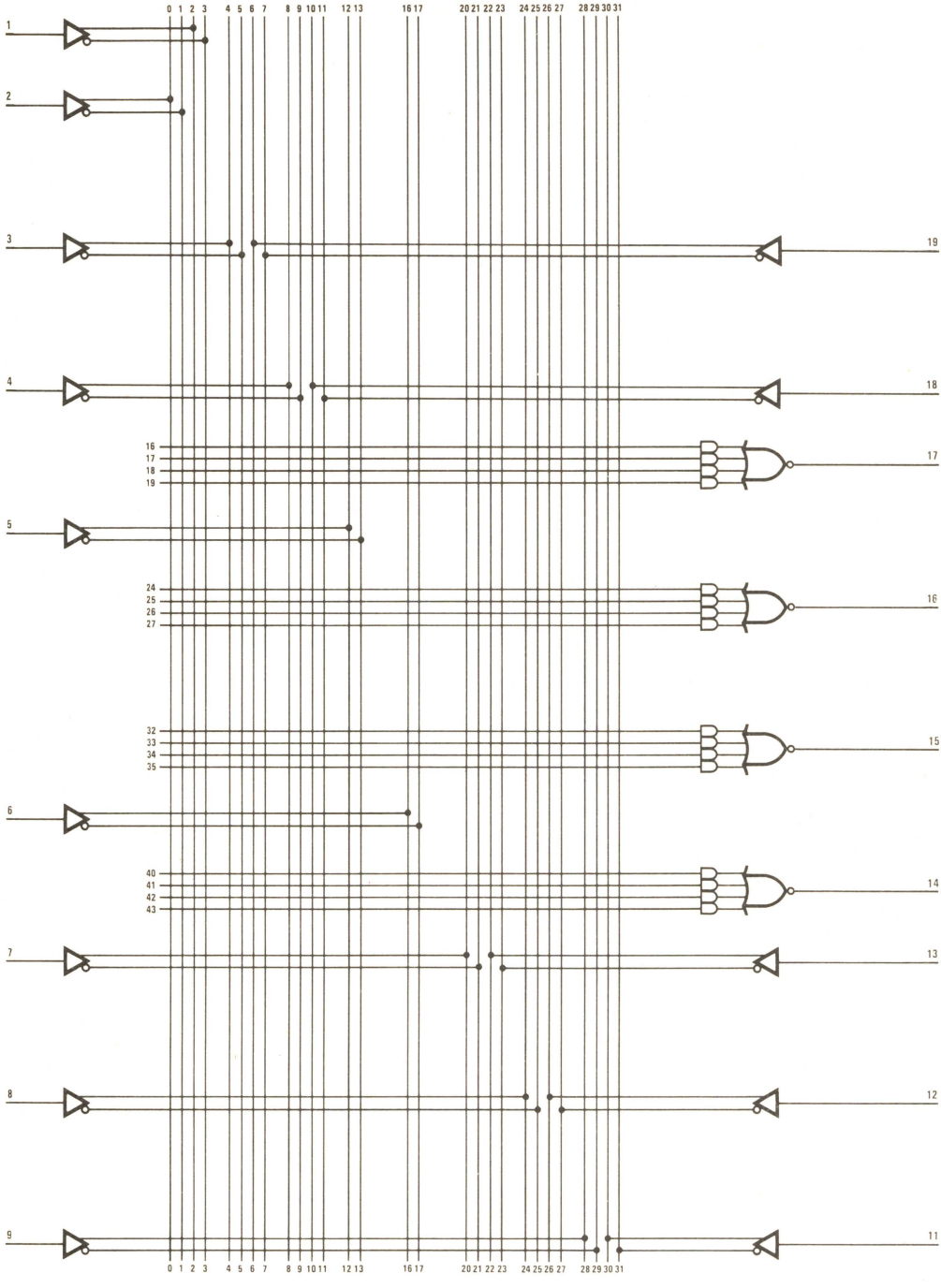
## Logic Diagram

### 12L6



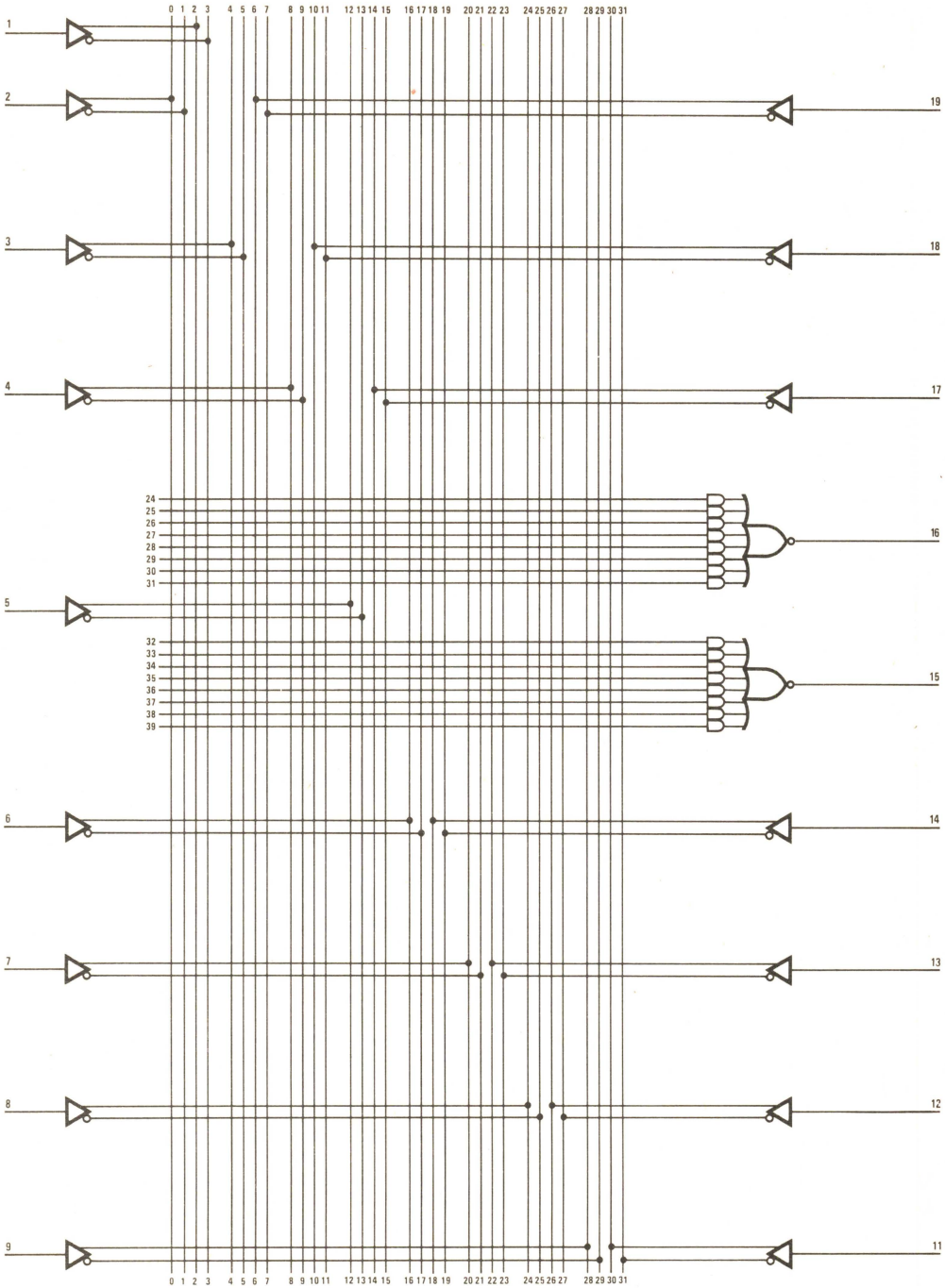
## Logic Diagram

### 14L4



## Logic Diagram

### 16L2



# High Speed Programmable Array Logic

# PAL32VX10
# PAL32VX10A

## Features/Benefits

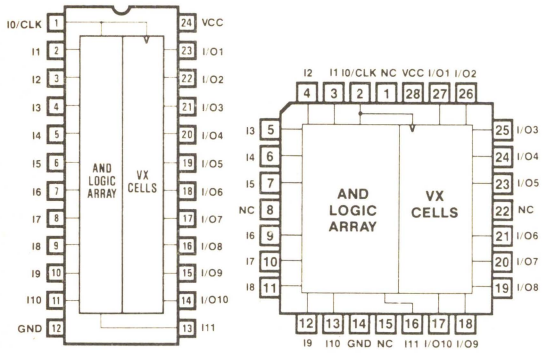
- Dual independent feedback paths allow buried state registers or input registers
- Programmable flip-flops allow J-K, S-R, T or D types for the most efficient use of product terms
- 10 input/output macrocells for flexibility
- Programmable registered or combinatorial outputs
- Programmable output polarity
- Global register asynchronous preset/synchronous reset or synchronous preset/asynchronous reset
- Automatic register preset on power up
- Preloadable output registers for testability
- Varied product term distribution
  —Up to 16 product terms per output
- High speed
  —25 ns "A" version
  —30 ns standard version
- Space-saving 24-pin 300-mil SKINNYDIP® package or 28-pin chip carrier
- Pin-compatible functional superset of 22V10

## Ordering Information

PAL32VX10A C NS STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE
  VX = Varied terms with XOR
- NUMBER OF OUTPUTS
- SPEED
  Blank = Standard
  A = High Speed
- PROCESSING
  STD = Standard
  XXXX = Other
- PACKAGE
  NS = Plastic SKINNYDIP
  JS = Ceramic SKINNYDIP
  FN = Plastic Leaded Chip Carrier
- TEMPERATURE RANGE
  C = 0°C to 75°C

## Pin Configurations




SKINNYDIP Package

Plastic Leaded Chip Carrier

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## General Description

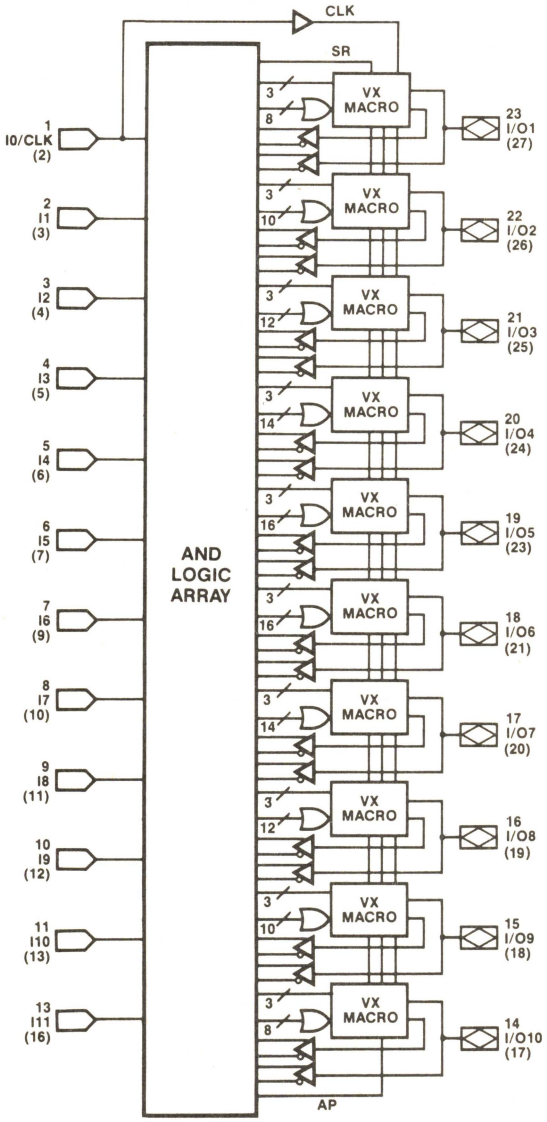
The PAL32VX10 is a high-density Programmable Array Logic (PAL®) device which implements a sum-of-products transfer function via a user-programmable AND logic array and a fixed OR logic array. Featured are ten highly flexible input/output macrocells which are user-configurable for combinatorial or registered operation. Each flip-flop can be programmed to be either a J-K, S-R, T, or D-type for optimal design of state machines and other synchronous logic. In addition, a unique dual feedback architecture allows I/O capability for each macrocell in both combinatorial and registered configurations. This can be achieved even when register feedback is present, and allows implementation of buried flip-flops while preserving the external macrocell input. Supplied in space-saving 300-mil-wide dual in-line packages or 28-pin chip carriers, the PAL32VX10 offers a powerful, space saving alternative to SSI/MSI logic devices, while providing the advantage of instant prototyping. Security fuses defeat readout after programming and make proprietary designs difficult to copy.

The PAL32VX10 is fabricated using Monolithic Memories' advanced oxide-isolated bipolar process for high speed and low power. TiW fuse links provide high reliability and programming yields. Special on-chip test circuits allow full AC, DC, and functional testing before programming. Preloadable output registers facilitate functional testing.

The PAL32VX10 can be programmed on standard PAL device programmers, fitted with appropriate programming modules and configuration software. Design development is supported by Monolithic Memories' PALASM® 2 software as well as by other programmable logic CAD tools available from third party vendors.

10290A
JANUARY 1988

## Block Diagram




Note: PLCC pin numbers are indicated in parentheses. PLCC pins 1, 8, 15, and 22 are not connected.

## Description of Architecture

The PAL32VX10 has twelve dedicated input lines and ten programmable I/O macrocells. Pin 1 serves either as an array input or as a clock for all flip-flops. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. The fuse matrix implements a programmable AND logic array, which drives a fixed OR logic array.

The high level of flexibility built into each macrocell, shown in Figure 1, allows the PAL32VX10 to implement over thirty different architecture options. Each macrocell can be individually programmed to implement a variety of combinatorial or registered logic functions.

## Dual Output Feedback

Dual feedback paths associated with each macrocell provide independent feedback paths directly into the array from both the flip-flop output and the output pin. Unlike other devices which have a single feedback path, the PAL32VX10 allows each output to have full I/O capability when configured as either a combinatorial output or a registered output, even if register feedback to the array is used. Thus registers can be loaded from their outputs.

If a macrocell is configured as a dedicated input, by disabling the three-state output buffer, the dual feedback architecture allows use of the associated register as an input register or as a "buried" state register, avoiding waste of the flip-flop, as shown in Figure 2.

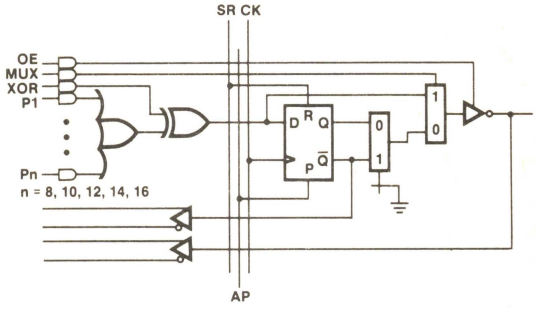



Figure 1. PAL32VX10 Macrocell

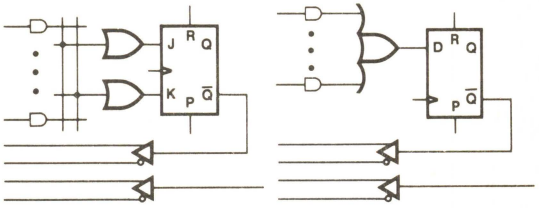



Figure 2. Buried Flip-Flops with Dedicated Inputs

5

## Programmable Flip-Flops

Each output macrocell contains a unique programmable flip-flop consisting of a basic D-type flip-flop driven by an XOR gate. This allows the user to choose the optimal flip-flop for the design, since either J-K, S-R, or T-type flip-flops can be synthesized from such a structure without wasting product terms.

As indicated in the macrocell logic diagram, one input of the XOR gate is connected to a single product term, while the second input is connected to the output of the OR logic array. The XOR gate output feeds the input of the D flip-flop. The way in which the XOR gate is used to synthesize the different flip-flop types is described in detail below.

**D Flip-Flop.** The D flip-flop option is implemented directly. In this configuration, the XOR gate on the input of the flip-flop can be used to program the logic polarity of the transfer function.

**J-K Flip-Flop.** The J-K flip-flop option can be easily synthesized with a more sophisticated manipulation of the XOR gate inputs and the D flip-flop output.

The transfer function of a J-K flip-flop can be mapped in the Karnaugh Map of Figure 3, where Q+ represents the next state of the flip-flop:

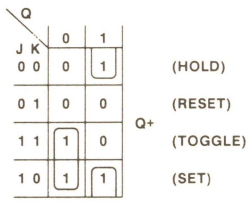



**Figure 3. J-K Flip-Flop Transfer Function**

Dropping the (+) for simplicity, the equivalent Boolean expression for Q+ is:

$$Q := \overline{K}*Q + J*\overline{Q}$$

In general, J and K can be sum-of-product expressions which are provided in the PAL architecture only in active-high form. Thus, a direct implementation of $\overline{K}$ expressions must invoke a DeMorgan transformation, which can use excessive product terms. This can be avoided by rewriting the equation for Q without inversions on the J or K inputs.

The XOR gate can be used to construct a logically equivalent expression without any inversions on the J or K inputs. The rewritten Boolean expression is:

$$Q := Q :+: (J*\overline{Q} + K*Q)$$

To check that these expressions are logically equivalent, change the XOR to its equivalent sum of products form (remember $A :+: B = A*\overline{B} + \overline{A}*B$) and reduce (using DeMorgan's theorem):

$Q := Q*\overline{(J*\overline{Q} + K*Q)} + \overline{Q}*(J*\overline{Q} + K*Q)$
$Q := Q*(\overline{(J + Q)}*\overline{(K + Q)}) + \overline{Q}*J*\overline{Q} + \overline{Q}*K*Q$
$Q := Q*(\overline{J}*\overline{K} + \overline{J}*\overline{Q} + Q*\overline{K} + Q*\overline{Q}) + J*\overline{Q}$
$Q := \overline{J}*\overline{K}*Q + \overline{K}*Q + J*\overline{Q}$

which simplifies to $Q := \overline{K}*Q + J*\overline{Q}$.

Since J and K are, in general, sums of products, J and K in either expression can be substituted with (J1 + J2 + ... + Jm) and (K1 + K2 + ... + Kn-m), where n is the total number of product terms associated with a given output macrocell. Thus, the total n-product term resource is shared between the J and K control inputs (Figure 4). Note that all J terms will contain $\overline{Q}$ and all K terms will contain Q.

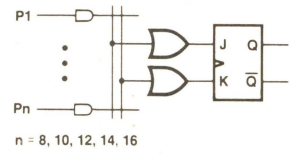



**Figure 4. J-K Flip-Flop Logic Equivalent; J and K Can Also be Active-Low**

The above discussions have assumed that it was most convenient to "group ones" in the Karnaugh Map. Sometimes it takes fewer product terms to "group zeros", i.e., implement the inversion of the desired function. The equations shown in Table 1 are equivalent and can be interchanged to optimize product term utilization. This can be readily proved through logic reductions similar to that above.

| | |
|---|---|
| J and K active high | $Q := Q :+: (J*\overline{Q} + K*Q)$ |
| J active high, K active low | $Q := J*\overline{Q} + \overline{K}*Q$ |
| J active low, K active high | $\overline{Q} := \overline{J}*\overline{Q} + K*Q$ |
| J and K active low | $Q := \overline{Q} :+: (\overline{J}*\overline{Q} + \overline{K}*Q)$ |

Note: J = sum of products J1 + J2 + ... + Jm
K = sum of products K1 + K2 + ... + Kn-m
n = total number of available product terms for a given macrocell (8 to 16)

**Table 1. J-K Flip-Flop Transfer Functions**

**S-R Flip-Flop.** The S-R flip-flop has a truth table identical to that of the J-K flip-flop, with the exception that the J=K=1 (toggle) condition is not allowed. The S-R flip-flop implementation is identical to that of the J-K flip-flop, with J-K replaced by S-R, and the S=R=1 condition avoided.

**T Flip-Flop.** A T (toggle) flip-flop either holds its state or toggles, depending on the logic state of the T input. The T flip-flop is a subset of the J-K flip-flop and can be considered equivalent to a J-K type with J=K. The general transfer function and its active-low T equivalent are both given in Table 2.

| |
|---|
| $Q := Q :+: T$ |
| $Q := \overline{Q} :+: \overline{T}$ |

Note: T = sum of products T1 + T2 + T3 + ... + Tn

**Table 2. T Flip-Flop Transfer Functions**

## Summary

The PAL32VX10 can synthesize J-K, S-R, T, and D flip-flops, whichever is most convenient for the application, without sacrificing product terms. Additionally, the synthesized equations can use the active-high or active-low forms of the inputs, allowing the designer to minimize product term requirements.

## Flip-Flop Bypass

Any output in the PAL32VX10 can be configured to be combinatorial by bypassing the output flip-flop. This is done by setting the output multiplexer to the appropriate state. The multiplexer is controlled by a product term which can be set unconditionally for a permanent combinatorial (all fuses opened, product term high) or registered (all fuses intact, product term low) output configuration, or can be programmed to bypass the output flip-flop "on the fly," allowing signals to be routed directly to output pins under user-specified conditions.

## Varied Product Term Distribution

An increased number of product terms has been provided in the PAL32VX10 over previous generation PAL devices. These terms are distributed among the ten macrocells in a varied manner, ranging from eight to sixteen terms per output. The five output pairs have 8, 10, 12, 14, or 16 product terms available for the OR gate within each macrocell. In addition, each macrocell has one XOR product term and two architecture control product terms.

## Programmable I/O

Each macrocell has a three-state output buffer with programmable three-state control. Control is implemented by a single product term, allowing specification of enable/disable functions controlled by any device input or output. Each macrocell can be configured as a dedicated input by disabling the buffer drive capability. When this is done, the associated register can still be used as an input register or buried state register, due to the independent register feedback path.

## Programmable Preset and Reset

The ten macrocell flip-flops share common programmable preset and reset control for easy system initialization. The Q outputs of the register will go to the logic low state following a low-to-high transition on pin 1 (I0/CLK) when the synchronous reset (SR) product term is asserted. The register will be forced to the logic high state independent of the clock when the asynchronous preset (AP) product term is asserted.

## Programmable Polarity

The polarity of each macrocell output can be set active high or active low.

**Combinatorial Outputs.** The XOR gate provides polarity control for combinatorial outputs, with the single product term to the XOR gate controlling the invert/not invert function. With all fuses intact, there is no inversion through the XOR gate, creating an active low output. Opening all fuses forces the product term high, inverting data and creating an active high output.

**Registered Outputs.** Output polarity for registered outputs can be determined in two ways. For D-type registered outputs, polarity can be set by the XOR gate, as is the case with combinatorial outputs. Using this method to set polarity, preset and reset will not be affected.

Polarity, as observed from the output pin, can also be determined by the flip-flop output multiplexer. Note that this does not affect the polarity of the register feedback signal, but does affect preset and reset. By changing the flip-flop output multiplexer, the preset and reset functions are exchanged, relative to the controlling product terms.

With the multiplexer fuse intact, the Q output is routed to the output pin, configuring an active low output. With the multiplexer fuse opened, $\overline{Q}$ is routed to the output pin, and synchronous reset becomes synchronous preset. Similarly, asynchronous reset becomes asynchronous preset.

Polarity options for J-K, S-R, and T flip-flops have been discussed in the section on programmable flip-flops.

## Power-Up Preset

All flip-flops power up to a logic high for predictable system initialization. Outputs of the PAL32VX10 will be high or low depending on the state of the register output multiplexers. See waveform at end of TTL/CMOS PAL Devices section.

## Register Preload

The register on the PAL32VX10 can be preloaded to facilitate functional testing of complex state machine designs. This feature allows direct loading of arbitrary states, thereby making it unnecessary to cycle through long test vector sequences to reach a desired state. In addition, transitions from illegal states can be verified by loading in illegal states and observing proper recovery.

## Security Fuse

After programming and verification, a PAL32VX10 design can be secured by programming the security fuses. Once programmed, these fuses defeat readback of the internal fuse pattern by a device programmer, making proprietary designs very difficult to copy.

## Quality and Testability

The PAL32VX10 offers a very high level of built-in quality. Special on-chip test circuitry provides a means of verifying performance of all AC and DC parameters prior to programming. In addition, these built-in test paths verify complete functionality of each device to provide the highest post-programming functional yields in the industry.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7 V | -0.5 V to 12 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 12 V |
| Off-state output voltage | 5.5 V | 12 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | STD | | | A | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 20 | 10 | | 18 | 10 | | ns |
| | | High | 20 | 10 | | 18 | 10 | | |
| $t_{su}$ | Setup time from input or feedback to clock | Product terms $P_1$-$P_n$, SR | 30 | 20 | | 25 | 20 | | ns |
| | | Product term XOR | 35 | 25 | | 30 | 25 | | |
| $t_h$ | Hold time | | 0 | -10 | | 0 | -10 | | ns |
| $t_{aw}$ | Asynchronous preset width | | 30 | 20 | | 25 | 20 | | ns |
| $t_{ar}$ | Asynchronous preset recovery time | | 30 | 20 | | 25 | 20 | | ns |
| $t_{sr}$ | Synchronous reset recovery time | | 30 | 20 | | 25 | 20 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 16 mA | | 0.35 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 140 | 180 | mA |
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 2.0 V at f = 1 MHz | | | 6 | | pF |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | 11 | | |

1. The PAL32VX10/A is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

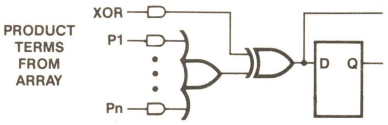
| SYMBOL | PARAMETER | | | TEST CONDITION | STD | | | A | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{PD}$ | Input or feedback to output | | Product terms $P_1$-$P_n$ | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 15 | 30 | | 10 | 25 | ns |
| | | | Product term XOR | | | 25 | 35 | | 20 | 30 | |
| $t_{CLK}$ | Clock to output or feedback | | | | | 10 | 15 | | 10 | 15 | ns |
| $t_{EA}$ | Input to output enable | | | | | 20 | 30 | | 20 | 25 | ns |
| $t_{ER}$ | Input to output disable | | | | | 20 | 30 | | 20 | 25 | ns |
| $t_{AP}$ | Asynchronous preset to output | | | | | 20 | 30 | | 20 | 25 | ns |
| $t_{CR}$ | Input or feedback to registered output from combinatorial configuration (Product term MUX 1→0) | | | | | 75 | 90 | | 75 | 90 | ns |
| $t_{RC}$ | Input or feedback to combinatorial output from registered configuration (Product term MUX 0→1) | | | | | 75 | 90 | | 75 | 90 | ns |
| $f_{MAX}$ | Maximum frequency | External | Product terms $P_1$-$P_n$ | | 22.5 | 35 | | 25 | 35 | | MHz |
| | | | Product term XOR | | 20 | 30 | | 22.2 | 30 | | |
| | | No feedback | | | 25 | 40 | | 27.7 | 40 | | |

## Use of XOR Product Term

The speed of the PAL32VX10 is specified according to the use of the Exclusive-OR XOR) product term in the macrocell. Note that the macrocell data input is a function of the two-input XOR gate, whose inputs are the OR of the product terms P1-Pn and the single additional XOR product term (Figure 1).

The specification for the path through the single XOR product term is 5 ns slower than through the P-1-Pn product terms and the OR gate. As a result, if the single XOR product term is changing, the macrocell data input will not be available until 5 ns later than if only the P1-Pn product terms were changing.

This difference between paths affects tPD, tsu, and fMAX (feedback). As a result, these three parameters are spedified both for only the P1-Pn product terms changing ("Product terms P1-Pn") and with the single XOR product term changing ("Product term XOR") (Figure 2).



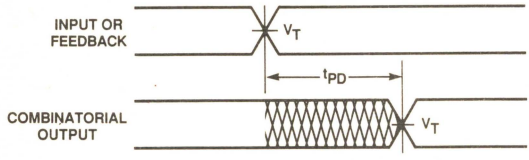

n = 8, 10, 12, 14, 16

Figure 1.

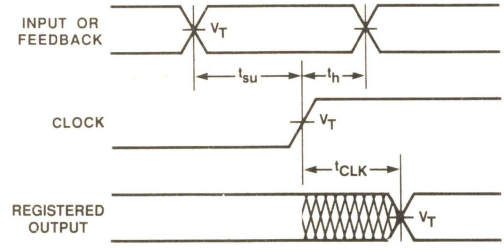
| SPECIFICATION | | EXPLANATION |
|---|---|---|
| $t_{PD}$, $t_{su}$, $f_{MAX}$ (feedback) | Product terms P1-Pn | If only the P1-Pn product terms are changing (XOR term is not changing) |
| | Product term XOR | If XOR term is changing |

Figure 2.

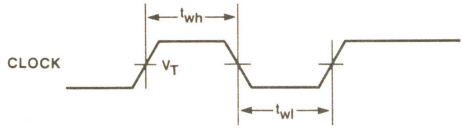
## Switching Waveforms



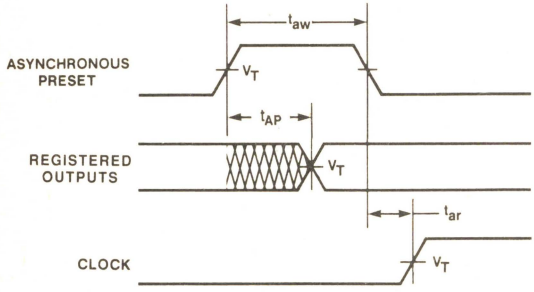

Combinatorial Output



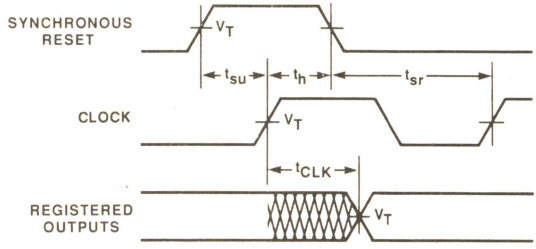

Registered Output




Clock Width

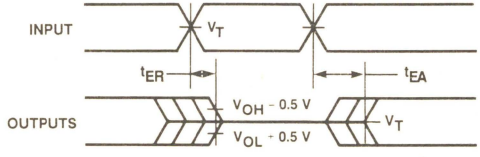



Asynchronous Preset




Synchronous Reset

Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.



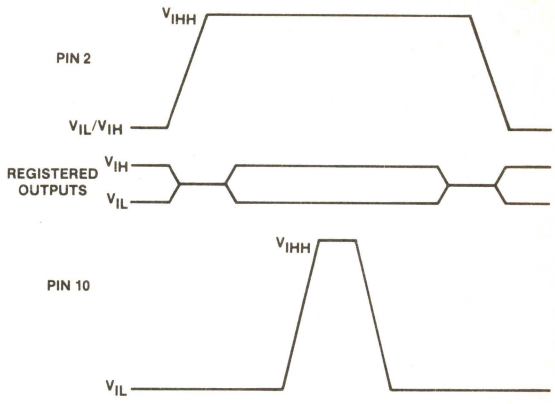

Input to Output Disable/Enable

## Output Register Preload

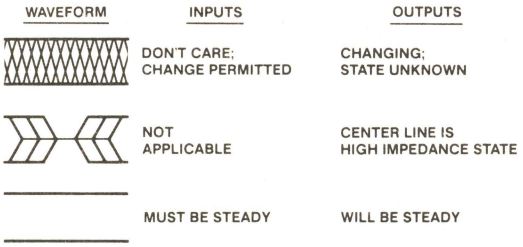
The preload function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct setting of output states. The procedure is:

1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 2 to $V_{IHH}$(12 V).
3. Apply $V_{IL}/V_{IH}$ to all registered output pins. Leave combinatorial outputs floating.
4. Pulse pin 10 to $V_{IHH}$, then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all output registers.
6. Remove high voltage from pin 2.
7. Enable registered outputs per programmed pattern.
8. Verify for $V_{OL}/V_{OH}$ at all registered output pins.

Note: $V_{IHH}$ = 11.0 (MIN), 11.5 (TYP) and 12.0 (MAX).




## Key to Timing Diagrams




5

## Switching Test Load

(refer to page 5-164)

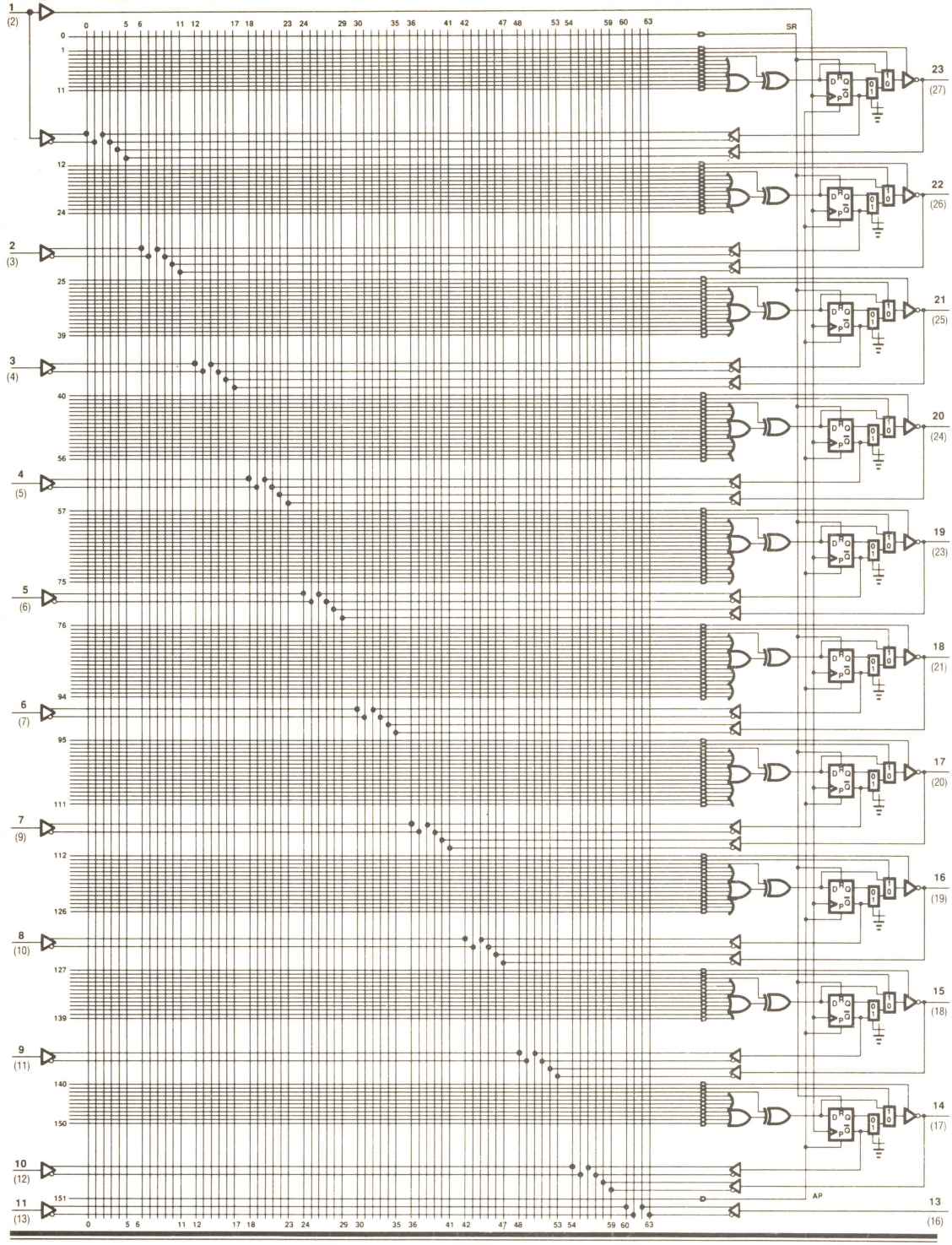
## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Schematic of Inputs and Outputs

(refer to page 5-164)

## Logic Diagram DIP (PLCC) Pinouts



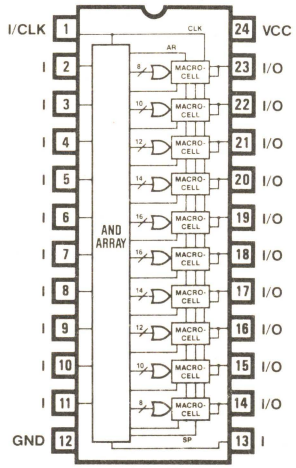
# CMOS Programmable Array Logic

# PALC22V10H-25
# PALC22V10H-35

## Features/Benefits

- CMOS technology cuts power in half (90 mA) while matching bipolar 22V10 speed
- 10 input/output macrocells for architectural flexibility
- Varied product term distribution
  - Up to 16 product terms per output
- Outputs programmable as registered or combinatorial
- Programmable output polarity
- Global register asynchronous reset and synchronous preset
- Preloadable output registers for testability
- Automatic register reset on power-up
- Erasable in windowed 24-pin SKINNYDIP® package
- Cost-effective OTP 24-pin SKINNYDIP packages and 28-pin Plastic Leaded Chip Carriers
- High-speed CMOS technology
  - 25 ns $t_{PD}$ for "-25" version
  - 35 ns $t_{PD}$ for "-35" version

## Pin Configurations




SKINNYDIP Package

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Description

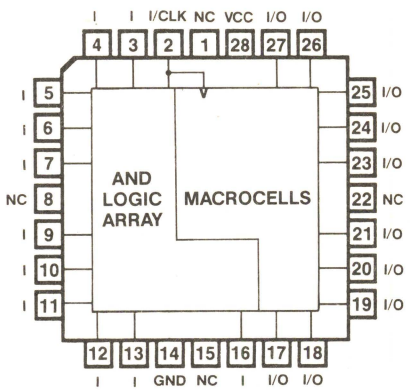
The PALC22V10 is an advanced PAL® device built with low-power CMOS technology. The device can be programmed to implement complex logic functions with up to twenty-two inputs and ten outputs. Design tools such as PALASM®2 software from Monolithic Memories allow automatic creation of a programming file based on the design description.

The PALC22V10 uses the familiar AND/OR logic array structure, which directly implements sum-of-products equations. The equations are programmed into the device through UV-erasable floating-gate cells in the AND logic array. The fixed OR logic array offers a varied number of product terms per output, with sixteen maximum. The sum of these products feeds the output macrocell. The macrocell can be programmed as registered or combinatorial, and active high or active low.

## Ordering Information

PALC22V10H-25 C NS STD

- PROGRAMMABLE ARRAY LOGIC
- CMOS TECHNOLOGY
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE: V = Varied Product Terms
- NUMBER OF OUTPUTS
- POWER: H = Half Power
- SPEED: -25 = 25 ns $t_{PD}$; -35 = 35 ns $t_{PD}$
- PROCESSING: STD = Standard; XXXX = Other
- PACKAGE: NS = Plastic SKINNYDIP; JS = Ceramic SKINNYDIP; QS = Windowed SKINNYDIP; FN = Plastic Leaded Chip Carrier
- OPERATING CONDITIONS: C = Commercial (0°C to 75°C)

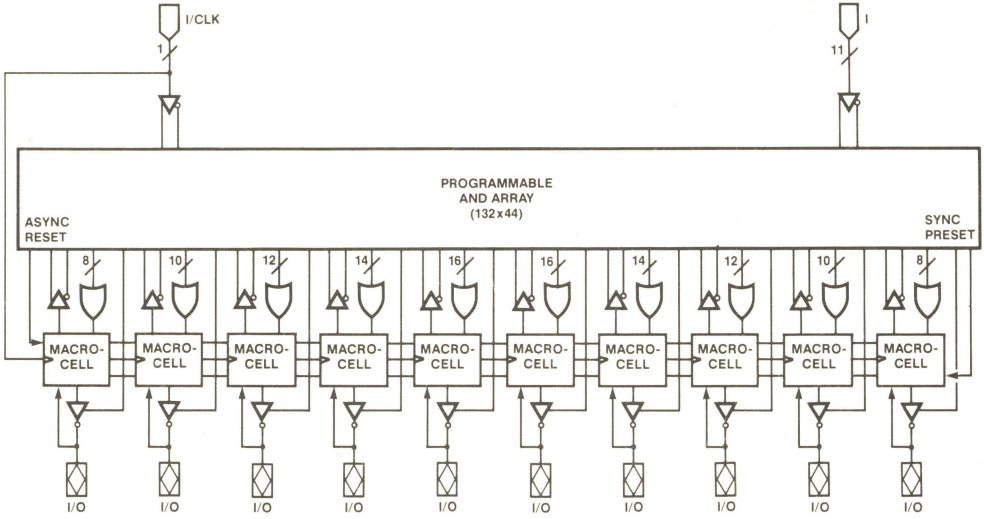



Plastic Leaded Chip Carrier

PAL®, PALASM® and SKINNYDIP® are registered trademarks of Monolithic Memories.

10293A
JANUARY 1988

## Block Diagram




## Architecture

The PALC22V10 has twelve dedicated input lines and ten programmable I/O macrocells. The macrocell is shown in Figure 1. Pin 1 serves either as an array input or as a clock for all flip-flops. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. The programming matrix implements a programmable AND logic array, which drives a fixed OR logic array.

## Macrocell

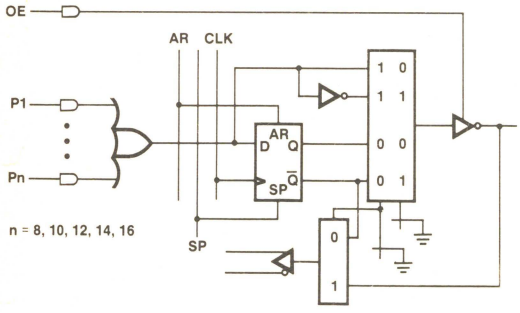



Figure 1.

The programmable functions in the PALC22V10 are automatically configured from the user's design specification, which can be in a number of formats. The design specification is processed by development software to verify the design and create a programming file. This file, once downloaded to a programmer, configures the device according to the user's desired function.

## Configuration Options

The output macrocell in the PALC22V10 allows four basic output configurations, as shown on the next page. The outputs can be either registered or combinatorial, and active high or active low, to match the needs of the design. Two programmable bits in each macrocell control a 4:1 output multiplexer and a 2:1 feedback multiplexer, selecting one of the four possible configurations for each output.

## Registered or Combinatorial Outputs

Each output of the PALC22V10 includes a D-type flip-flop for data storage and synchronization. Any output can be configured to be combinatorial by selecting a path that bypasses the output flip-flop. Bypass is controlled by one of two programmable bits on the output multiplexer, and is automatically selected if requested in the design specification. The unprogrammed state is a registered output configuration. The registered configuration includes register feedback, while the combinatorial configuration includes I/O feedback.

## Programmable I/O

Each macrocell has a three-state output buffer with programmable three-state control. A product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The macrocell provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

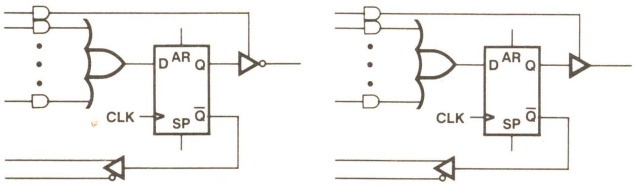
## Programmable Polarity

The polarity of each macrocell output can be active high or active low, either to match output signal needs or to reduce product terms. Selection is controlled by the second of two programmable bits in the output macrocell, and affects both registered and combinatorial outputs. The unprogrammed configuration is active low. Selection is automatically performed according to the design specification and pin definitions.

Note that preset and reset control the flip-flop, not the output. Thus, if active low polarity is selected, the effects of preset and reset on the output will be exchanged.

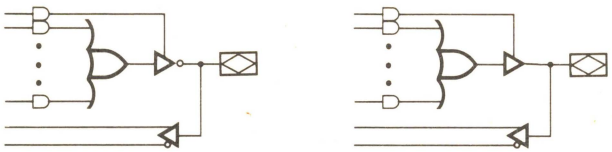
## Configuration Options

Registered Outputs




ACTIVE LOW ACTIVE HIGH

Combinatorial I/O



ACTIVE LOW ACTIVE HIGH

## Varied Product Term Distribution

An increased number of product terms has been provided in the PALC22V10 over previous generation PAL devices, increasing the logic capabilities of the device. These product terms are distributed among the ten macrocells in a varied manner, ranging from eight to sixteen terms per output. The varied distribution allows optimum use of device resources. The outputs have 8, 10, 12, 14, or 16 product terms available for the OR gate within each macrocell.

## Programmable Preset and Reset

The ten macrocell flip-flops share common programmable preset and reset control for easy system initialization. The Q outputs of the register will go to the logic high state following a low-to-high transition of pin 1 (I/CLK) when the synchronous preset (SP) product term is asserted. The register will be forced to the logic low state independent of the clock when the asynchronous reset (AR) product term is asserted. Product term control allows preset and reset to be functions of any combination of device inputs and output feedback. The outputs will be high or low depending upon the polarity option chosen.

## Power-Up Reset

All flip-flops power up to a logic low for predictable system initialization. Outputs of the PALC22V10 will be high or low depending on whether the output is active low or active high, respectively. The VCC rise must be monotonic, and the reset delay time is 1 µs maximum.

## Register Preload

The register on the PALC22V10 can be preloaded from the output pins to facilitate functional testing of complex state machine designs. This feature allows direct loading of arbitrary states, thereby making it unnecessary to cycle through long test vector sequences to reach a desired state. In addition, transitions from illegal states can be verified by loading illegal states and observing proper recovery.

## Security Bit

After programming and verification, a PALC22V10 design can be secured by programming the security bit. Once programmed, this bit defeats readback of the internal programmed pattern by a device programmer, securing proprietary designs.

## Quality and Testability

The PALC22V10 offers a very high level of built-in quality. Extra programmable bits and the erasability of the device provide a means of verifying performance of all AC and DC parameters. In addition, this verifies complete programmability and functionality of the device to provide the highest programming yields and post-programming functional yields in the industry.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 5.25 V |
| Input voltage | -3.0 V to 7.0 V | -1.0 V to 14.0 V |
| Off-state output voltage | -0.5 V to 7.0 V | -0.5 V to 7.0 V |
| DC Output current into outputs | | 16 mA |
| Storage temperature | | -65° C to +150° C |
| Ambient temperature with power applied | | -55° C to +125° C |
| UV light exposure | | 7258 W-sec/cm$^2$ |
| Static discharge voltage | | >2001 V |
| Latchup current ($T_A$ = 0° C to 75° C) | | >100 mA |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | -35 | | | -25 | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.5 | 5 | 5.5 | V |
| $t_{wl}$ | Width of clock | Low | 17 | 14 | | 13 | 9 | | ns |
| $t_{wh}$ | | High | 17 | 14 | | 13 | 9 | | |
| $t_{su}$ | Setup time from input, feedback, or SP to clock | | 25 | 20 | | 15 | 11 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | 0 | -10 | | ns |
| $t_{aw}$ | Asynchronous reset width | | 35 | 30 | | 25 | 20 | | ns |
| $t_{ar}$ | Asynchronous reset recovery time | | 35 | 30 | | 25 | 20 | | ns |
| $t_{sr}$ | Synchronous reset recovery time | | 35 | 30 | | 25 | 20 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2.0 | | | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | | -10 | μA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 10 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 10 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 16 mA | | 0.35 | 0.4 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -40 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 40 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = MAX | $V_O$ = 0.5 V | -30 | -70 | -90 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX, $V_I$ = GND, Outputs open | | | 60 | 90 | mA |
| $C_{IN}$ | Input capacitance[5] | $V_{IN}$ = 2.0 V at f = 1 MHz | | | | 5 | pF |
| $C_{OUT}$ | Output capacitance[5] | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | | 8 | pF |

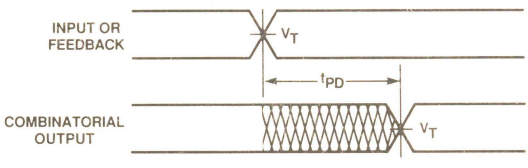
1. The PALC22V10 is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time and duration of the short-circuit should not exceed one second.
5. Sampled but not 100% tested.

## Switching Characteristics Over Operating Conditions

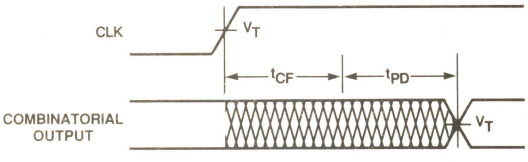
| SYMBOL | PARAMETER | | TEST CONDITIONS | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | -35 | | | -25 | | | |
| | | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{PD}$ | Input or feedback to output | | $R_1$ = 300 Ω<br>$R_2$ = 390 Ω | | 25 | 35 | | 20 | 25 | ns |
| $t_{CLK}$ | CLK to output | | | | 20 | 25 | | 13 | 15 | ns |
| $t_{CF}$[1] | CLK to feedback | | | | 12 | 18 | | 9 | 13 | ns |
| $t_{EA}$ | Input to output enable | | | | 30 | 35 | | 20 | 25 | ns |
| $t_{ER}$ | Input to output disable | | | | 30 | 35 | | 20 | 25 | ns |
| $t_{ARO}$ | Asynchronous reset to output | | | | 25 | 35 | | 20 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External feedback ($1/t_{su} + t_{CLK}$) | | 20 | 25 | | 33.3 | 41 | | MHz |
| | | Internal feedback ($1/t_{su} + t_{CF}$) | | 23 | 30 | | 35 | 50 | | |
| | | No feedback ($1/t_{twh} + t_{wl}$) | | 29 | 35 | | 38 | 55 | | |

1. Calculated from measured clock to feedback to combinatorial output minus $t_{PD}$.

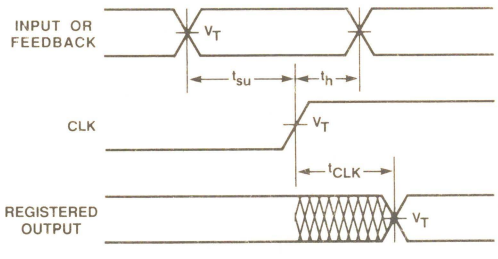
## Switching Waveforms



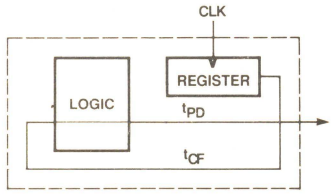

Combinatorial Output

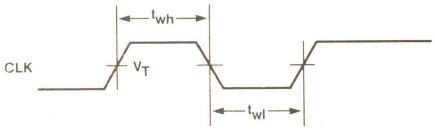



Clock to Feedback to Combinatorial Output (see path below)



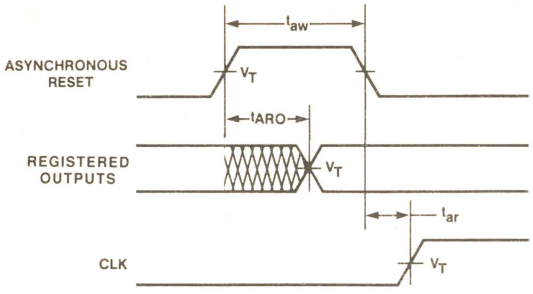

Registered Output



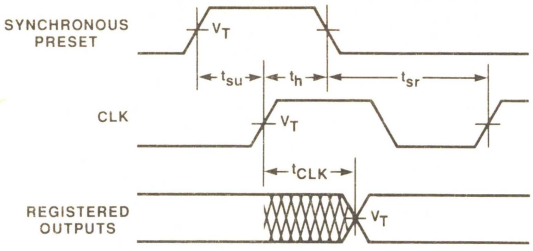




Clock Width

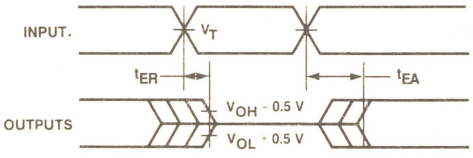
5




Asynchronous Reset




Synchronous Preset



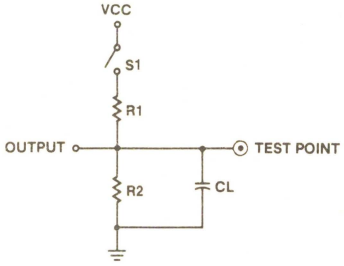

Input to Output Disable/Enable

Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.

## Key to Timing Diagrams

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE; CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

## Switching Test Load

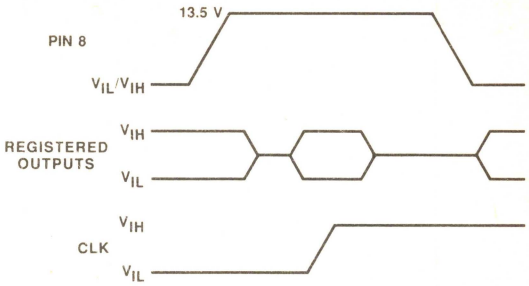



| SPECIFICATION | SWITCH S1 | $C_L$ | MEASURED OUTPUT VALUE |
|---|---|---|---|
| $t_{PD}$, $t_{CLK}$, $t_{CF}$ | closed | 50 pF | 1.5 V |
| $t_{EA}$ | Z->H: open<br>Z->L: closed | 50 pF | 1.5 V |
| $t_{ER}$ | H->Z: open<br>L->Z: closed | 5 pF | H->Z: $V_{OH}$-0.5 V<br>L->Z: $V_{OL}$+0.5 V |

## Output Register Preload

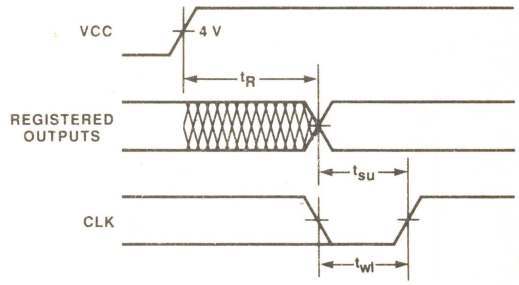
The preload function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct loading of output states. The procedure is:

1. Raise VCC to 5.0 V ± 0.5 V.
2. Disable output registers by setting pin 8 (DIP) to 13.5 V ± 0.5 V.
3. Apply VIL/VIH as desired to all registered output pins. Leave combinatorial outputs floating.
4. Clock output registers.
5. Remove VIL/VIH from all registered output pins.
6. Remove high voltage from pin 8.
7. Enable output registers per programmed pattern.
8. Verify for VOL/VOH at all registered output pins, according to programmed polarity.




## Power-Up Reset

All flip-flops power up to a logic low for predictable system initialization. The power-up reset time, tR, is 1µs maximum. The required setup and clock widths are listed in the specifications. The outputs will be high or low, depending on the polarity option chosen.




## Programming and Erasing

The PALC22V10 can be programmed on standard logic programmers. Programmers approved by Monolithic Memories are listed on the following page. The PALC22V10 may be erased by ultraviolet light when contained in the windowed package.

For erasure, the recommended ultraviolet light wavelength is 2537 Angstroms. The minimum dose required is 72000 mW-sec/cm$^2$ (UV intensity x exposure time). For an ultraviolet lamp with a 20 mW/cm$^2$ power rating, the minimum exposure time would be 72000/20 seconds, or 60 minutes. The device needs to be within 1 inch of the lamp during erasure.

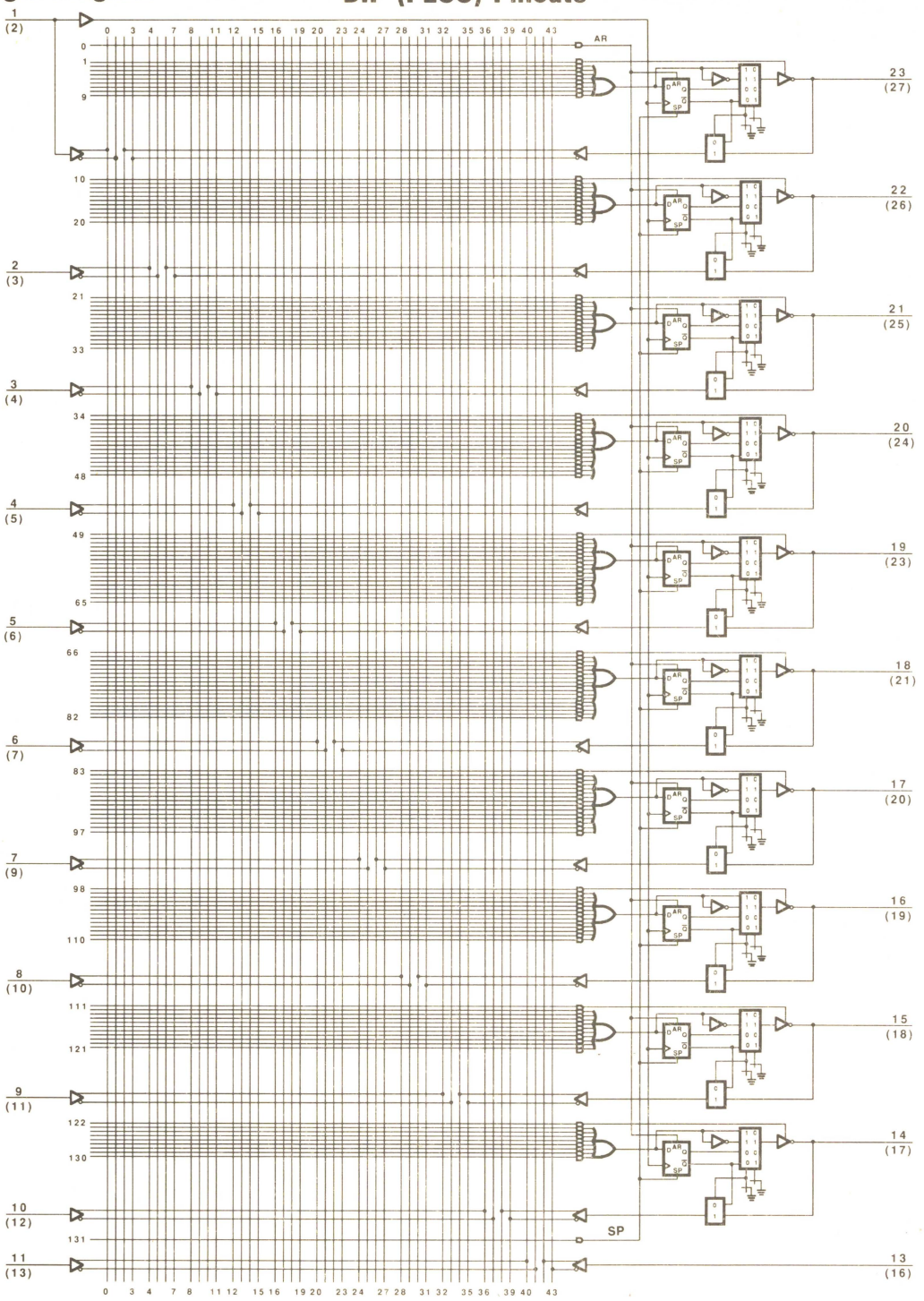
Permanent damage may result if the device is exposed to high intensity UV light for an extended period of time. The recommended maximum dosage is 7258 W-sec/cm$^2$.

Wavelengths of light less than 4000 Angstroms can partially erase the device in the windowed package. For this reason, an opaque label should be placed over the window, especially if the device will be exposed to sunlight or fluorescent lighting for extended periods of time.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Logic Diagram



# High Speed Programmable Array Logic

# PAL22RX8A

## Features/Benefits

- **Programmable flip-flops allow J-K, S-R, T or D-types for the most efficient use of product terms**
- **8 input/output macrocells for flexibility**
- **Programmable registered or combinatorial outputs**
- **Programmable output polarity**
- **Global register asynchronous preset/asynchronous reset**
- **Automatic register reset on power up**
- **Preloadable output registers for testability**
- **High speed at 25 ns $t_{PD}$, 28.5 MHz $f_{MAX}$**
- **Space-saving 24-pin 300-mil SKINNYDIP® package or 28-pin chip carrier**

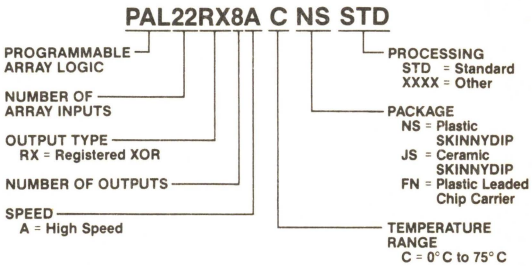
## General Description

The PAL22RX8A is a high-density Programmable Array Logic (PAL®) device which implements a sum-of-products transfer function via a user-programmable AND logic array and a fixed OR logic array. Featured are eight highly flexible input/output macrocells which are user-configurable for combinatorial or registered operation. Each flip-flop can be programmed to be either a J-K, S-R, T, or D-type for optimal design of state machines and other synchronous logic. The PAL22RX8A is a functional superset of, and pin-compatible with, the PAL24A Series, but can implement many new functions due to its added features. Supplied in space-saving 300-mil-wide dual in-line packages or 28-pin chip carriers, the PAL22RX8A offers a powerful, space-saving alternative to SSI/MSI logic devices, while providing the advantage of instant prototyping. Security fuses defeat readout after programming and make proprietary designs difficult to copy.

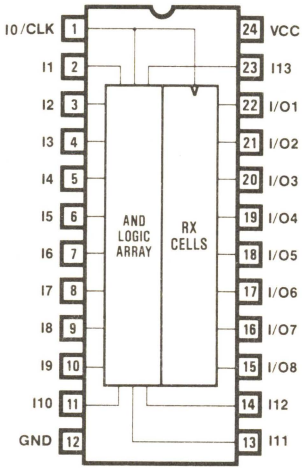
The PAL22RX8A is fabricated using Monolithic Memories' advanced bipolar process for high speed and low power. TiW fuse links provide high reliability and programming yields. Special on-chip test circuits allow full AC, DC, and functional testing before programming. Preloadable output registers facilitate functional testing.

The PAL22RX8A can be programmed on standard PAL device programmers, fitted with appropriate programming modules and configuration software. Design development is supported by Monolithic Memories' PALASM® 2 software as well as by other programmable logic CAD tools available from third party vendors.

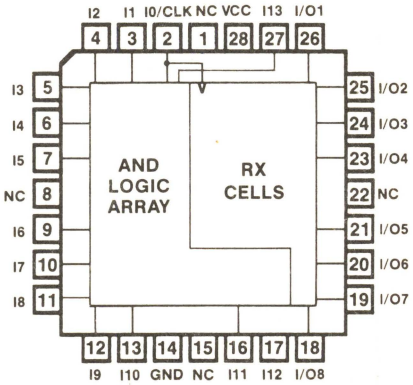
## Ordering Information




## Pin Configurations




SKINNYDIP Package

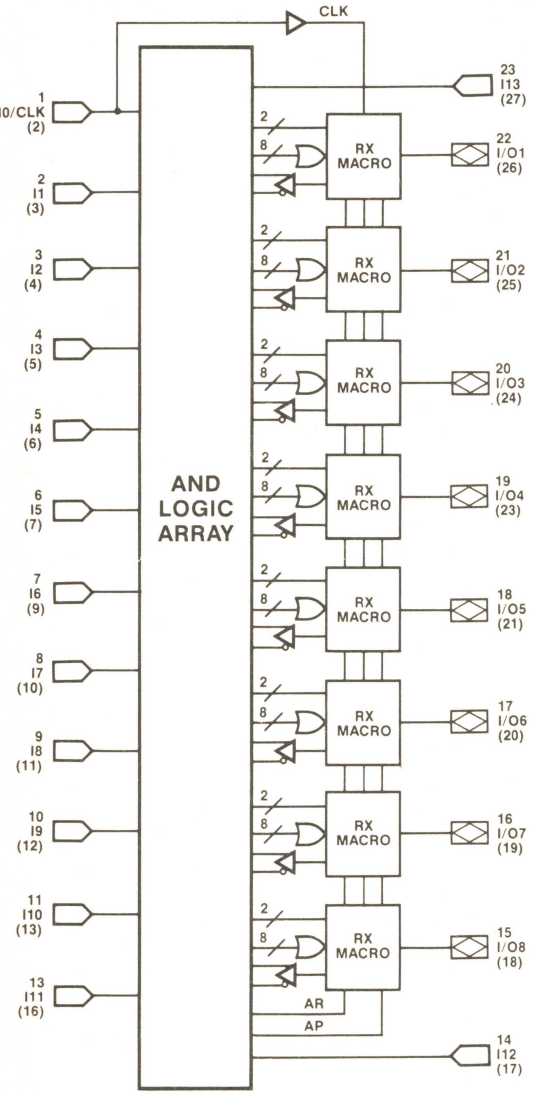



Plastic Leaded Chip Carrier

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Block Diagram




Note:
PLCC pin numbers are indicated in parentheses.
PLCC pins 1, 8, 15 and 22 are not connected.

## Description of Architecture

The PAL22RX8A has fourteen dedicated inputs and eight programmable I/O macrocells. Pin 1 serves either as an array input or as a clock for all flip-flops. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. The fuse matrix implements a programmable AND logic array, which drives a fixed OR logic array.

The high level of flexibility built into each macrocell, shown in Figure 1, allows the PAL22RX8A to implement several different architecture options. Each macrocell can be individually programmed to implement a variety of combinatorial or registered logic functions.

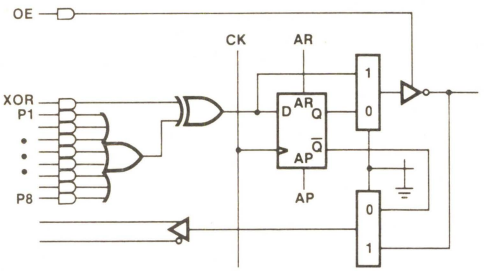



Figure 1. PAL22RX8A Macrocell

## Programmable Flip-Flops

Each output macrocell contains a unique programmable flip-flop consisting of a basic D-type flip-flop driven by an XOR gate. This allows the user to choose the optimal flip-flop for the design, since either J-K, S-R, or T-type flip-flops can be synthesized from such a structure without wasting product terms.

As indicated in the macrocell logic diagram, one input of the XOR gate is connected to a single product term, while the second input is connected to the output of the OR logic array. The XOR gate output feeds the input of the D flip-flop. The way in which the XOR gate is used to synthesize the different flip-flop types is described in detail below.

**D Flip-Flop.** The D flip-flop option is implemented directly. In this configuration, the XOR gate on the input of the flip-flop can be used to program the logic polarity of the transfer function.

**J-K Flip-Flop.** The J-K flip-flop option can be easily synthesized with a more sophisticated manipulation of the XOR gate inputs and the D flip-flop output.

The transfer function of a J-K flip-flop can be mapped in the Karnaugh Map of Figure 2, where Q+ represents the next state of the flip-flop:

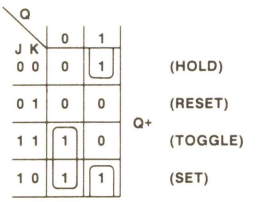



Figure 2. J-K Flip-Flop Transfer Function

Dropping the (+) for simplicity, the equivalent Boolean expression for Q+ is:

$$Q := \overline{K}*Q + J*\overline{Q}$$

In general, J and K can be sum-of-product expressions which are provided in the PAL architecture only in active-high form. Thus, a direct implementation of $\overline{K}$ expressions must invoke a DeMorgan transformation, which can use excessive product terms. This can be avoided by rewriting the equation for Q without inversions on the J or K inputs.

The XOR gate can be used to construct a logically equivalent expression without any inversions on the J or K inputs. The rewritten Boolean expression is:

$$Q := Q :+: (J*\overline{Q} + K*Q)$$

To check that these expressions are logically equivalent, change the XOR to its equivalent sum of products form (remember $A :+: B = A*\overline{B} + \overline{A}*B$) and reduce (using DeMorgan's theorem):

$$Q := Q*\overline{(J*\overline{Q} + K*Q)} + \overline{Q}*(J*\overline{Q} + K*Q)$$
$$Q := Q*(\overline{J} + Q)*(\overline{K} + \overline{Q}) + \overline{Q}*J*\overline{Q} + \overline{Q}*K*Q$$
$$Q := Q*(\overline{J}*\overline{K} + J*\overline{Q} + Q*\overline{K} + Q*\overline{Q}) + J*\overline{Q}$$
$$Q := \overline{J}*\overline{K}*Q + \overline{K}*Q + J*\overline{Q}$$

which simplifies to $Q := \overline{K}*Q + J*\overline{Q}$.

Since J and K are, in general, sums of products, J and K in either expression can be substituted with (J1 + J2 + ... + Jm) and (K1 + K2 + ... + K8-m), where 8 is the total number of product terms associated with a given output macrocell. Thus, the total 8 product term resource is shared between the J and K control inputs (Figure 3). Note that all J terms will contain $\overline{Q}$ and all K terms will contain Q.

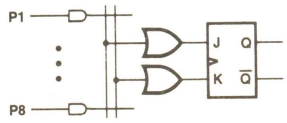



**Figure 3. J-K Flip-Flop Logic Equivalent; J and K Can Also be Active-Low**

The above discussions have assumed that it was most convenient to "group ones" in the Karnaugh Map. Sometimes it takes fewer product terms to "group zeros", i.e., implement the inversion of the desired function. The equations shown in Table 1 are equivalent and can be interchanged to optimize product term utilization. This can be readily proved through logic reductions similar to that above.

| | |
|---|---|
| J and K active high | $Q := Q :+: (J*\overline{Q} + K*Q)$ |
| J active high, K active low | $Q := J*\overline{Q} + \overline{K}*Q$ |
| J active low, K active high | $\overline{Q} := \overline{J}*\overline{Q} + K*Q$ |
| J and K active low | $Q := \overline{Q} :+: (\overline{J}*\overline{Q} + \overline{K}*Q)$ |

Note: J = sum of products J1 + J2 + ... + Jm
K = sum of products K1 + K2 + ... + K8-m
8 = total number of available product terms for a given macrocell

**Table 1. J-K Flip-Flop Transfer Functions**

**S-R Flip-Flop.** The S-R flip-flop has a truth table identical to that of the J-K flip-flop, with the exception that the J=K=1 (toggle) condition is not allowed. The S-R flip-flop implementation is identical to that of the J-K flip-flop, with J-K replaced by S-R, and the S=R=1 condition avoided.

**T Flip-Flop.** A T (toggle) flip-flop either holds its state or toggles, depending on the logic state of the T input. The T flip-flop is a subset of the J-K flip-flop and can be considered equivalent to a J-K type with J=K. The general transfer function and its active-low T equivalent are both given in Table 2.

| |
|---|
| $Q := Q :+: T$ |
| $Q := \overline{Q} :+: \overline{T}$ |

Note: T = sum of products T1 + T2 + T3 + ... + T8.

**Table 2. T Flip-Flop Transfer Functions**

## Summary

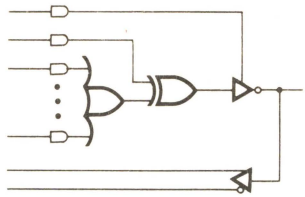
The PAL22RX8A can synthesize J-K, S-R, T, and D flip-flops, whichever is most convenient for the application, without sacrificing product terms. Additionally, the synthesized equations can use the active-high or active-low forms of the inputs, allowing the designer to minimize product term requirements.

5

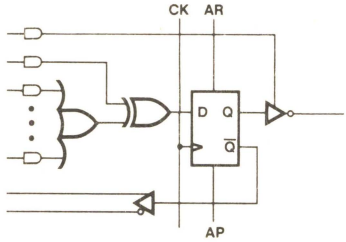
## Flip-Flop Bypass

Any output in the PAL22RX8A can be configured as combinatorial by bypassing the output flip-flop. This is done by setting the output multiplexer to the appropriate state. The multiplexer is controlled by a single architecture fuse which, when intact, selects a registered output with register feedback and, when opened, selects a combinatorial output with feedback from the output pin.

Combinatorial I/O



Registered Output




## Programmable I/O

Each macrocell has a three-state output buffer with programmable three-state control. Control is implemented by a single product term, allowing specification of enable/disable functions controlled by any device input or output. Each macrocell can be configured as a dedicated input by selecting the combinatorial output configuration and disabling the buffer drive capability.

## Programmable Polarity

The polarity of each macrocell output can be set active high or active low.

**Combinatorial Outputs.** The XOR gate provides polarity control for combinatorial outputs, with the single product term to the XOR gate controlling the invert/not invert function. With all fuses intact, there is no inversion through the XOR gate, creating an active high output.

**Registered Outputs.** For D-type registered outputs, polarity can be set by the XOR gate, as is the case with combinatorial outputs. Using this method to set polarity, preset and reset will not be affected.

Polarity options for J-K, S-R, and T flip-flops have been discussed in the section on programmable flip-flops.

## Programmable Preset and Reset

The eight macrocell flip-flops share common programmable preset and reset control for easy system initialization. The Q outputs of the register will go to the logic low state independent of the clock when the asynchronous reset (AR) product term is asserted. The register will be forced to the logic high state independent of the clock when the asynchronous preset (AP) product term is asserted.

## Power-Up Reset

All flip-flops power up to a logic low for predictable system initialization. The power-up state of the flip-flop is not affected by the output polarity option chosen. Due to the output inverter, the outputs will power up to a logic high. See waveform at end of TTL/CMOS PAL Devices section.

## Register Preload

The register on the PAL22RX8A can be preloaded by use of supervoltages to facilitate functional testing of complex state machine designs. This feature allows direct loading of arbitrary states, thereby making it unnecessary to cycle through long test vector sequences to reach a desired state. In addition, transitions from illegal states can be verified by preloading illegal states and observing proper recovery.

## Security Fuses

After programming and verification, a PAL22RX8A design can be secured by programming the security fuses. Once programmed, these fuses defeat readback of the internal fuse pattern by a device programmer, making proprietary designs very difficult to copy.

## Quality and Testability

The PAL22RX8A offers a very high level of built-in quality. Special on-chip test circuitry provides a means of verifying performance of all AC and DC parameters prior to programming. In addition, these built-in test paths verify complete functionality of each device to provide the highest post-programming functional yields in the industry.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7 V | −0.5 V to 12 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 12 V |
| Off-state output voltage | 5.5 V | 12 V |
| Storage temperature | | −65°C to +150°C |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 10 | 7 | | ns |
| | | High | 10 | 7 | | |
| $t_{su}$ | Setup time from input or feedback to clock | | 20 | 16 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | ns |
| $t_{aw}$ | Asynchronous preset/reset width | | 15 | 10 | | ns |
| $t_{ar}$ | Asynchronous preset/reset recovery time | | 30 | 25 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITION | | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | | 2.0 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | | −0.1 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | | 0.35 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | | 180 | 210 | mA |
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 2.0 V at f = 1 MHz | | DIP pins 1, 13 | | 15 | | pF |
| | | | | All other inputs | | 12 | | |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | | 12 | | |

Notes: 1. The PAL22RX8A is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.

2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.

3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).

4. No more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

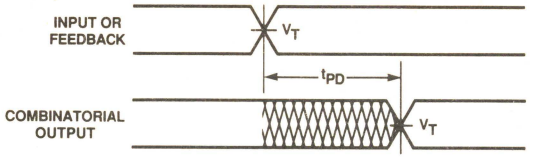
| SYMBOL | PARAMETER | | TEST CONDITIONS | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 20 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | 12 | 15 | ns |
| $t_{EA}$ | Input to output enable | | | | 20 | 25 | ns |
| $t_{ER}$ | Input to output disable | | | | 18 | 25 | ns |
| $t_{AP}$ | Asynchronous preset/reset to output | | | | 25 | 35 | ns |
| $f_{MAX}$ | Maximum frequency | External feedback | | 28.5 | 40 | | MHz |
| | | No feedback | | 50 | 55 | | |

### Switching Test Load
(refer to page 5-164)

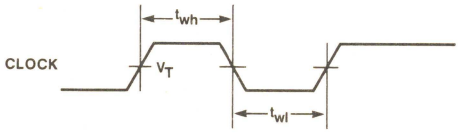
### Power-Up Reset Waveform
(refer to page 5-164)

### Schematic of Inputs and Outputs
(refer to page 5-164)

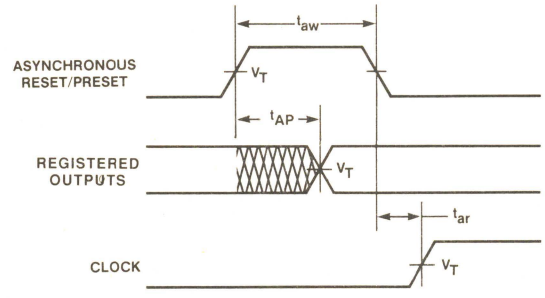
## Switching Waveforms



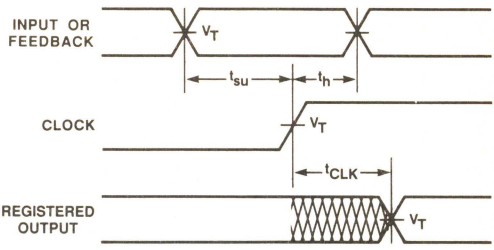

Combinatorial Output



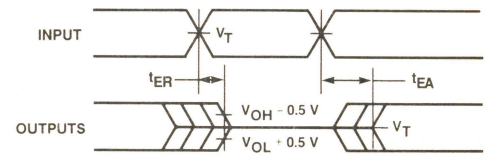

Clock Width




Asynchronous Reset/Preset




Registered Output




Input to Output Disable/Enable

Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

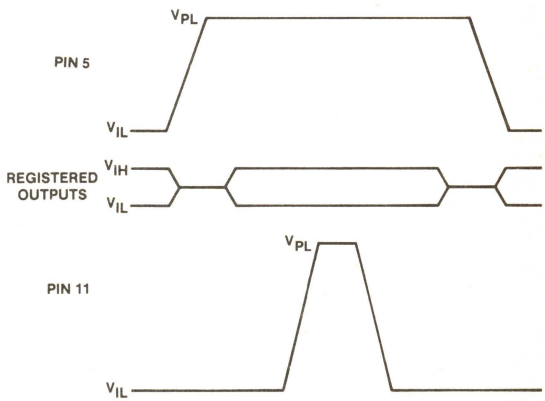
## Output Register Preload

The preload function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct setting of output states. The procedure is:

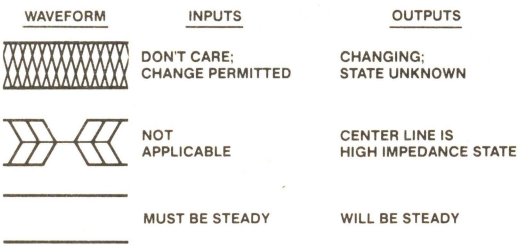
1. Raise $V_{CC}$ to 5.0 V.
2. Set pin 1 (CLK) to $V_{IL}$.
3. Disable output registers by setting pin 5 to $V_{PL}$ (18.0 V ± 0.5 V).
4. Apply the desired level ($V_{IL}/V_{IH}$) to all registered output pins. Leave combinatorial outputs floating.
5. Pulse pin 11 to $V_{PL}$, then back to 0 V.
6. Remove $V_{IL}/V_{IH}$ from all output registers.
7. Lower pin 5 to $V_{IL}$.
8. Enable output registers per programmed pattern.
9. Verify for $V_{OL}/V_{OH}$ at all registered output pins.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)



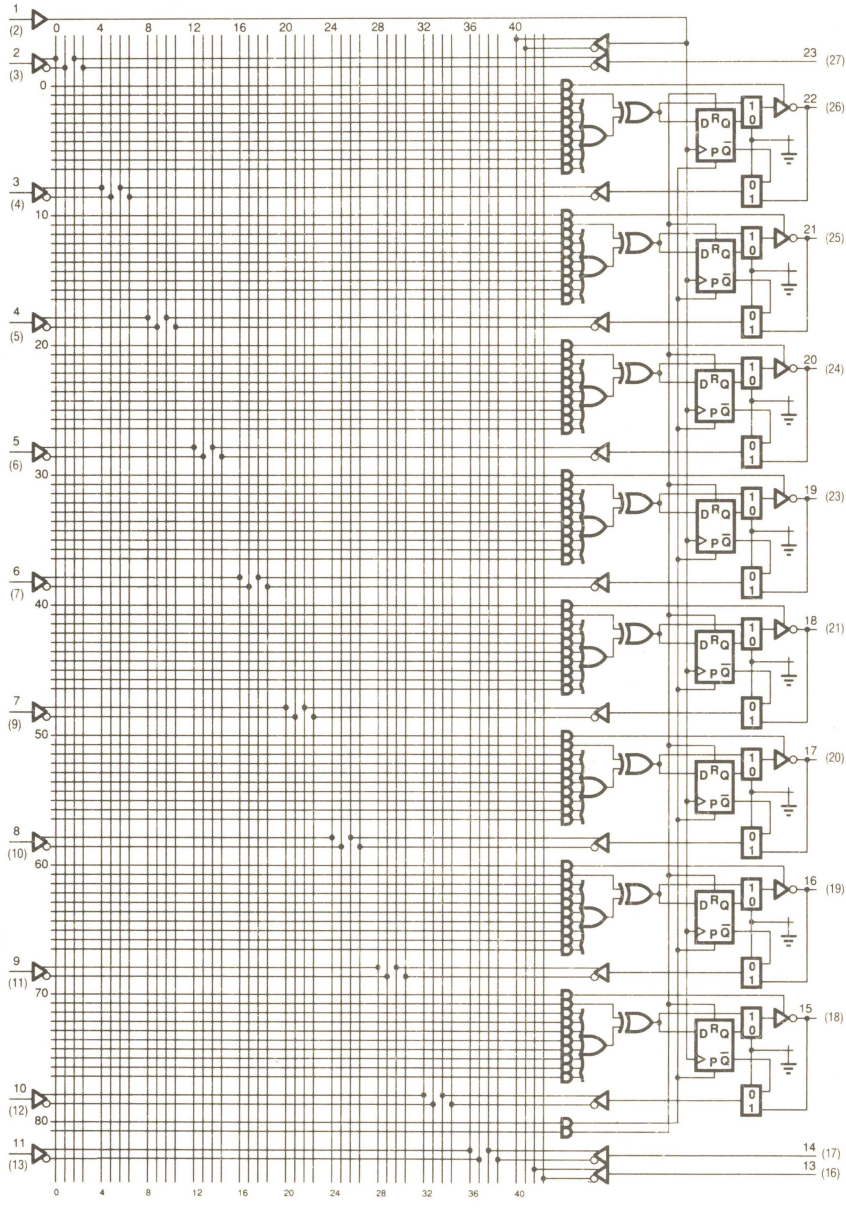

## Key to Timing Diagrams



## Logic Diagram

## DIP (PLCC) Pinouts

PAL22RX8A



## ADVANCE INFORMATION

### Features/Benefits

- High-speed 24-pin PAL® device with asynchronous flip-flop control
- 20 ns propagation delay (active low)
- Programmable clock allows independent clocking of each flip-flop
- Programmable asynchronous set and reset for each flip-flop
- Programmable output polarity
- Programmable flip-flop bypass allows any output to be combinatorial
- Three-state outputs controlled by both product term and dedicated pin for flexibility
- Power-up reset for automatic initialization
- TTL-level register preload simplifies functional testing
- Easy design with PALASM®2 software

### Description

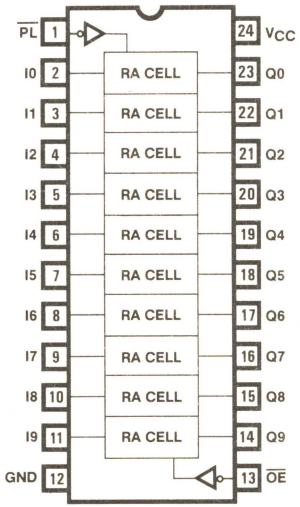
The PAL20RA10 is an improved-speed version of the Registered Asynchronous PAL device. This device offers independent asynchronous controls for each input/output macrocell. The 24-pin PAL20RA10 has ten inputs and ten I/O macrocells. The macrocell is described in detail on the next page.

The high-speed oxide-isolated bipolar technology provides very high performance. Maximum propagation delay is 20 ns for active low outputs (polarity fuse unprogrammed), and 25 ns for active high outputs (polarity fuse programmed).

Design is accomplished using PALASM 2 software from Monolithic Memories, or third-party development programs. Boolean equations are automatically converted into a programming pattern. The pattern, once downloaded to any PLD programmer, can instantly provide a custom device. The programmed pattern can be protected from copying by programming the security fuse.

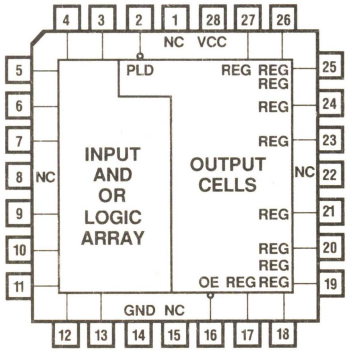
### Pin Configurations

PAL20RA10-20




DIP

PAL20RA10-20CFN

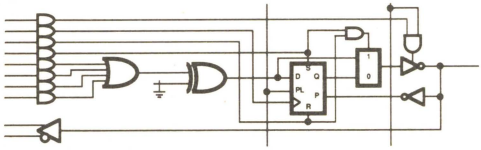



Plastic Leaded Chip Carrier

5

## Macrocell

The macrocell offers programmable clock, enable, set, and reset for each flip-flop, each of which can operate asynchronously with respect to other flip-flops in the same device. These features effectively provide independent 7474-type flip-flops preceded by the PAL logic functions. Each macrocell also provides independent flip-flop bypass, allowing any combination of combinatorial and registered outputs in a single device.



## Programmable Clock

The clock input to each flip-flop comes from the programmable array, allowing any flip-flop to be clocked independently if desired.

## Programmable Set and Reset

Each flip-flop has a product line for asynchronous set and one product for asynchronous reset. If the chosen product line is HIGH, the flip-flop will set (become a logic HIGH), or reset (become a logic LOW). The sense of the output pin is inverted.

## Programmable Flip-Flop Bypass

If both the set and reset product lines are HIGH, the flip-flop is bypassed and the output becomes combinatorial. Thus each output can be configured to be combinatorial or registered.

## Register Preload

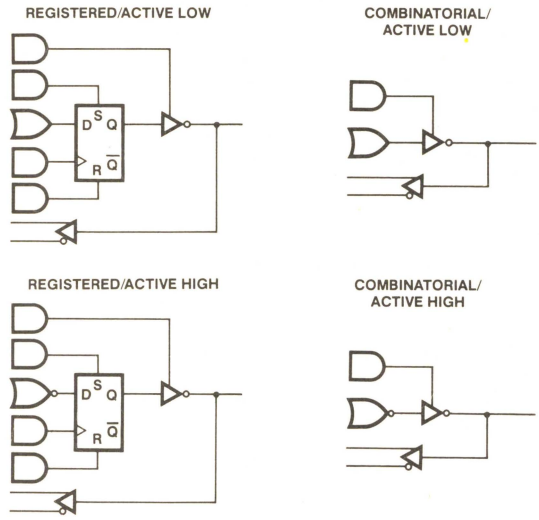
Both devices offer register preload for device testability. The registers can be preloaded from the outputs by using TTL-level signals, in order to simplify functional testing.

## Power-Up Reset

The devices also offer power-up reset. On the application of power to the VCC pin, the flip-flops will reset to a logic LOW. Because of the output inverter, the output pins will be a logic HIGH on power-up.

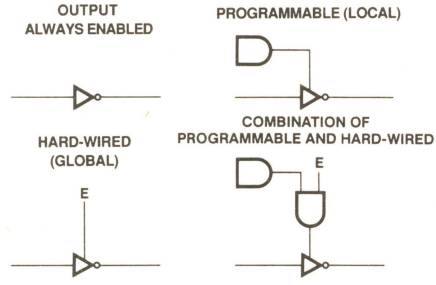
## Programmable Polarity

Each flip-flop input has individually programmable polarity. The unprogrammed state is active LOW.




## Three-State Outputs

The devices provide a product term dedicated to local output control. There is also a global output control pin. The output is enabled if both the global output control pin is LOW and the local output control product term is HIGH. If the global output control pin is HIGH, all outputs will be disabled. If a local output control product term is LOW, then that output will be disabled.



# Asynchronous PAL20RA10

## Features/Benefits

- Programmable clock for asynchronous operation
- Programmable asynchronous set and reset
- Programmable polarity
- Programmable flip-flop bypass
- Local and global output enable control
- TTL level register preload
- Power-up reset
- Complements 20-pin PAL16RA8
- High speed, as fast as 20 ns tPD for PAL20RA10-20 Series
- Security fuse on all devices

## Ordering Information

PAL20RA10-20 C NS STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- REGISTERED ASYNCHRONOUS
- NUMBER OF OUTPUTS
- SPEED
  - None = Standard (30 ns)
  - -20 = 20 ns
- OPERATING CONDITIONS
  - C = Commercial
- PROCESSING
  - STD = Standard
  - XXXX = Other
- PACKAGE
  - NS = Plastic SKINNYDIP
  - JS = Ceramic SKINNYDIP
  - NL = Plastic Leaded Chip Carrier
  - FN = Plastic Leaded Chip Carrier (Center No-connects)
  - SG = Small-Outline Gull-wing

## Description

The PAL20RA10 is a 24-pin registered asynchronous PAL device. This versatile device features programmable clock, enable, set, and reset, all of which can operate asynchronously to other flip-flops in the same device. It also has individual flip-flip bypass, allowing this one device to provide any combination of registered and combinatorial outputs.

## Programmable Clock

The clock input to each flip-flop comes from the programmable array, allowing the flip-flops to be clocked independently if desired.

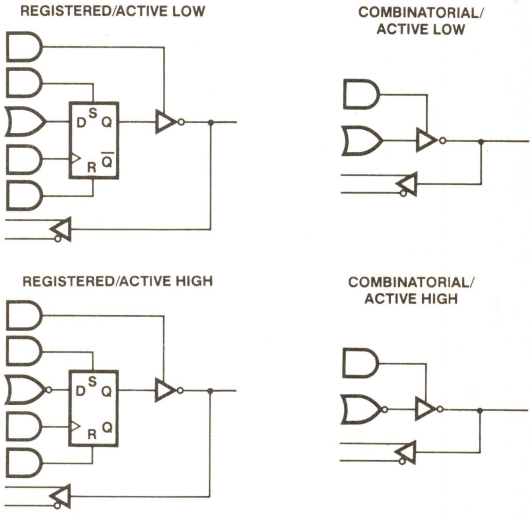
## Programmable Set and Reset

Each flip-flop has a product line for asynchronous set and one product for asynchronous reset. If the chosen product line is high, the flip-flop will set (become a logic HIGH), or reset (become a logic LOW). The sense of the output pin is inverted if the output is active low.

## Programmable Polarity

Each flip-flop has individually programmable polarity. The unprogrammed state is active low.

## Programmable Output Polarity




5

## Programmable Flip-Flop Bypass

If both the set and reset product lines are high, the flip-flop is bypassed and the output becomes combinatorial. Thus each output can be configured to be registered or combinatorial.

## Programmable and Hard-Wired Three-State Outputs

The PAL20RA10 provides a product term dedicated to output control. There is also an output control pin (pin 13). The output is enabled if both the output control pin is low and the output control product term is high. If the output control pin is high, all outputs will be disabled. If an output control product term is low, then that output will be disabled.

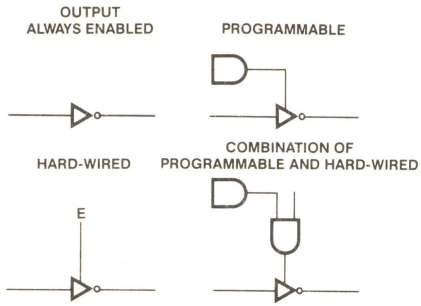
## Register Preload and Power-Up Reset

Each device also offers register preload for device testability. The registers can be preloaded from the outputs by using TTL level signals in order to simplify functional testing. This series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

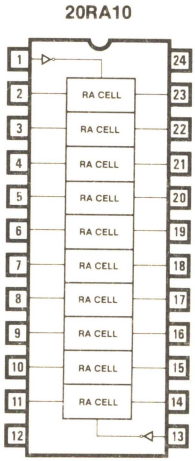
## Packages

The commercial PAL20RA10 Series is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), plastic leaded chip carrier (NL), and small outline (SG) packages.

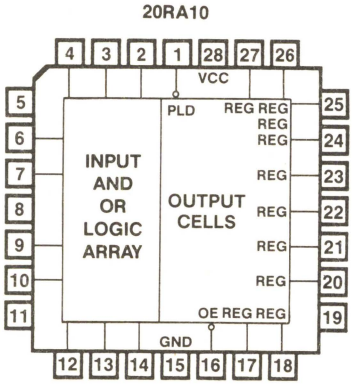
## Output Control Alternatives




## DIP/SO Pinouts




## PLCC Pinout




## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 20 | 13 | | ns |
| | | High | 20 | 13 | | |
| $t_{wp}$ | Preload pulse width | | 35 | 15 | | ns |
| $t_{su}$ | Set up time from input or feedback to clock | | 20 | 10 | | ns |
| $t_{sup}$ | Preload set up time | | 25 | 5 | | ns |
| $t_h$ | Hold time | Polarity fuse intact | 10 | −2 | | ns |
| | | Polarity fuse programmed | 0 | −6 | | |
| $t_{hp}$ | Preload hold time | | 25 | 5 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

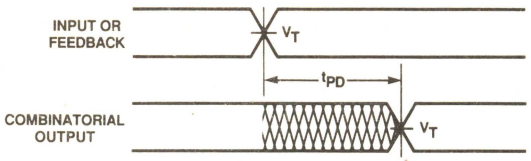
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = 3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 155 | 200 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

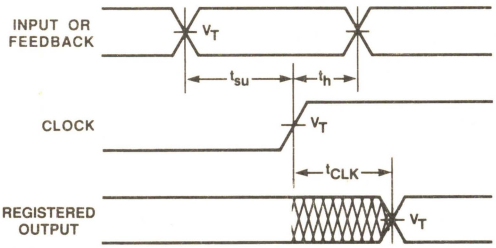
## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | Polarity fuse intact | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | 20 | 30 | ns |
| | | Polarity fuse programmed | | | 25 | 35 | |
| $t_{CLK}$ | Clock to output or feedback | | | 10 | 17 | 30 | ns |
| $t_S$ | Input to asynchronous set | | | | 22 | 35 | ns |
| $t_R$ | Input to asynchronous reset | | | | 27 | 40 | ns |
| $t_{PZX}$ | Pin 13 to output enable | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 13 to output disable | | | | 10 | 20 | ns |
| $t_{EA}$ | Input to output enable | | | | 18 | 30 | ns |
| $t_{ER}$ | Input to output disable | | | | 15 | 30 | ns |
| $f_{MAX}$ | Maximum frequency | External | | 20 | 35 | | MHz |
| | | No feedback | | 25 | 38 | | |

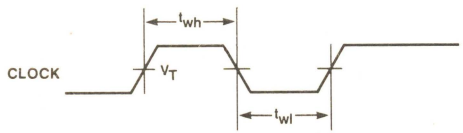
## Switching Waveforms



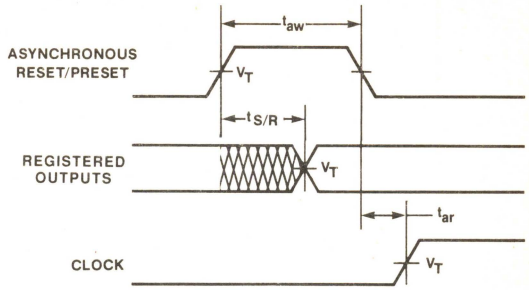

Combinatorial Output



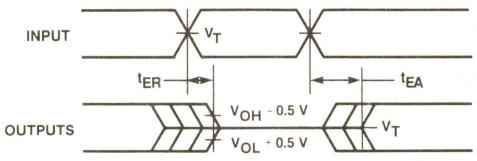

Registered Output



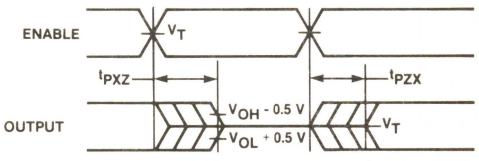

Clock Width




Asynchronous Reset/Preset




Input to Output Disable/Enable




Pin 13 to Output Disable/Enable

Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagrams

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | DON'T CARE; CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | NOT APPLICABLE | CENTER LINE IS HIGH IMPEDANCE STATE |
| | MUST BE STEADY | WILL BE STEADY |

## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Power-Up Reset Waveform

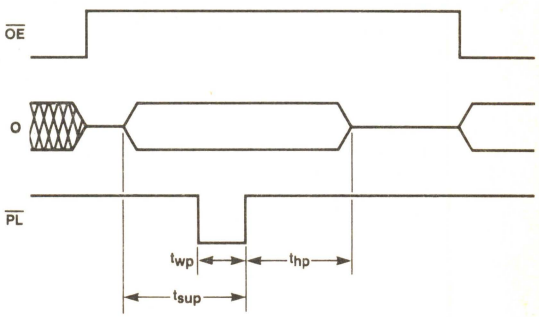
(refer to page 5-164)

## Schematic of Inputs and Outputs

(refer to page 5-164)

## Register Preload

Register preload allows any arbitrary state to be loaded into the PAL device output registers. This allows complete logic verification, including states that are impossible or impractical to reach. To use the preload feature, first disable the outputs by bringing $\overline{OE}$ high, and present the data at the output pins. A low level on the preload pin ($\overline{PL}$) will then load the data into the registers.

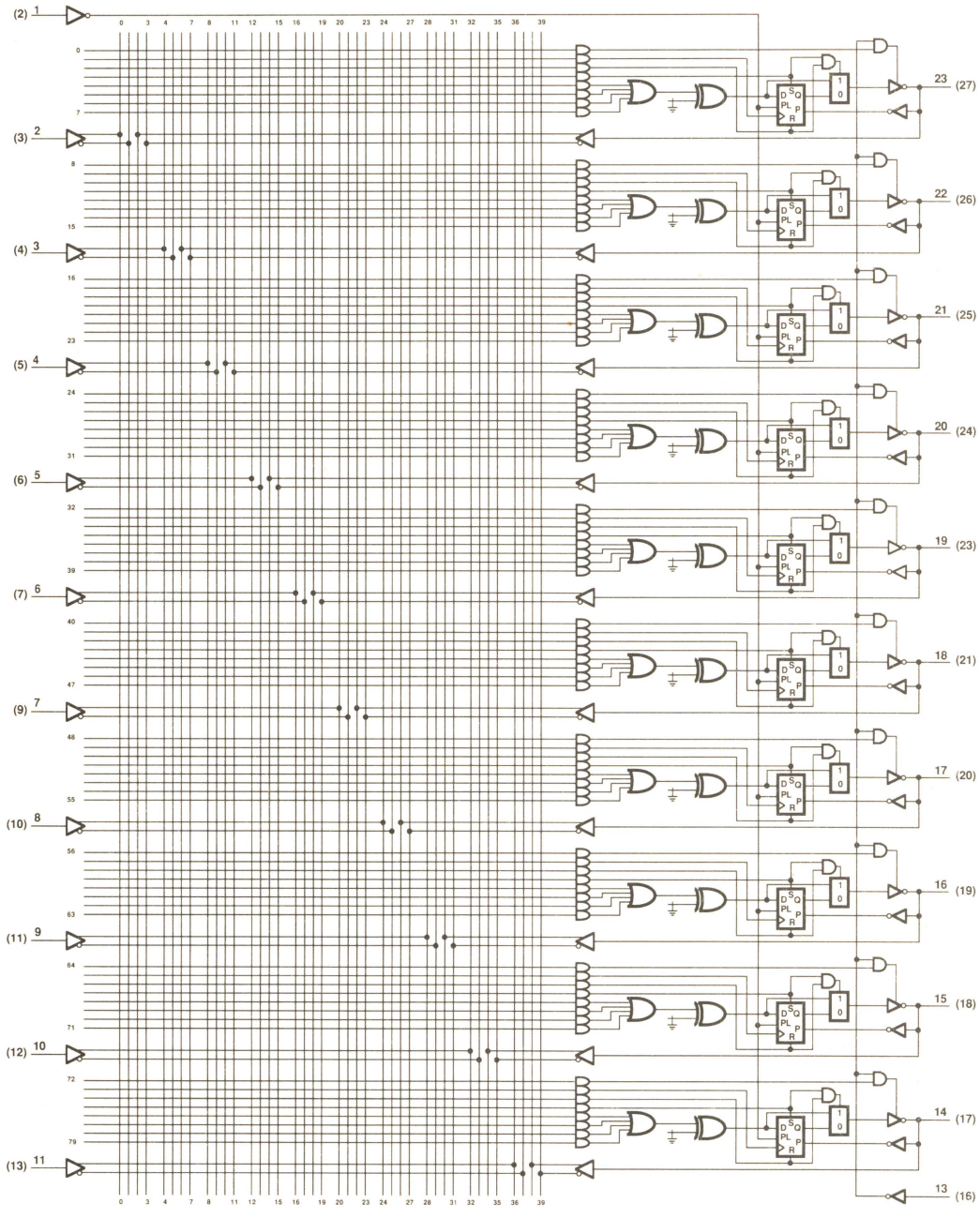



5

## Logic Diagram

DIP (PLCC) Pinouts

PAL20RA10



# PAL20RS10 Series

# 20S10, 20RS10 20RS8, 20RS4

## Features/Benefits

- Product term steering allows up to 16 product terms per output
- Programmable polarity
- Register preload
- Power-up reset
- Security fuse

## Ordering Information

PAL20RS10 C NS STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE
  - RS = Registered with Product Term Steering
  - S = Combinatorial with Product Term Steering
- NUMBER OF OUTPUTS
- OPERATING CONDITIONS
  - C = Commercial
- PACKAGE
  - NS = Plastic SKINNYDIP
  - JS = Ceramic SKINNYDIP
  - NL = Plastic Leaded Chip Carrier
  - SG = Small-Outline Gull-wing
- PROCESSING
  - STD = Standard
  - XXXX = Other

## PAL20RS10 Series

| | ARRAY INPUTS | OUTPUTS | | $t_{PD}$* (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|---|
| | | COMBINATORIAL | REGISTERED | | |
| PAL20S10 | 20 | 10 | 0 | 35/40 | 240 |
| PAL20RS10 | 20 | 0 | 10 | 35 | 240 |
| PAL20RS8 | 20 | 2 | 8 | 35/40 | 240 |
| PAL20RS4 | 20 | 6 | 4 | 35/40 | 240 |

*35 ns active low, 40 ns active high

## Description

The PAL20RS10 Series offers product term steering, which allows up to sixteen product terms to be used at a single output.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

The registered devices have ten dedicated input lines, and each combinatorial output is an I/O pin. The combinatorial device has twelve dedicated input lines, and only eight of the ten combinatorial outputs are I/O pins. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)



10301A
JANUARY 1988

## Programmable Polarity

Each flip-flop has individually programmable polarity. The unprogrammed state is active low.

## Product Term Steering

Product term steering allows each pair of outputs to share its product terms with one output or the other (not both). Each pair has a total of sixteen product terms; thus, one output can use zero to sixteen terms while the other has sixteen to zero. Product terms can only be shared mutually exclusively. If both outputs need the same term, it must be created twice, once for each output.

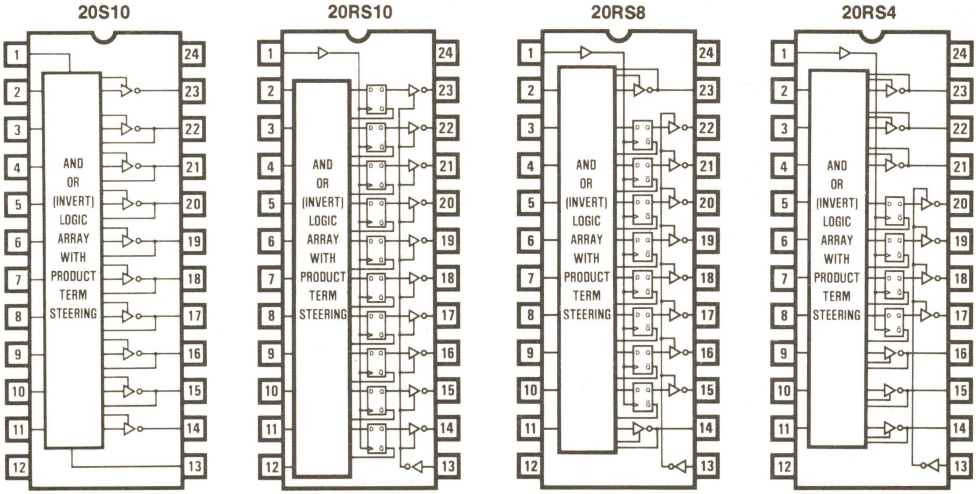
## Preload and Power-Up Reset

The 20RS10 Series offers register preload for device testability. The registers can be preloaded from the outputs by using super-voltages in order to simplify functional testing. The 20RS10 Series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

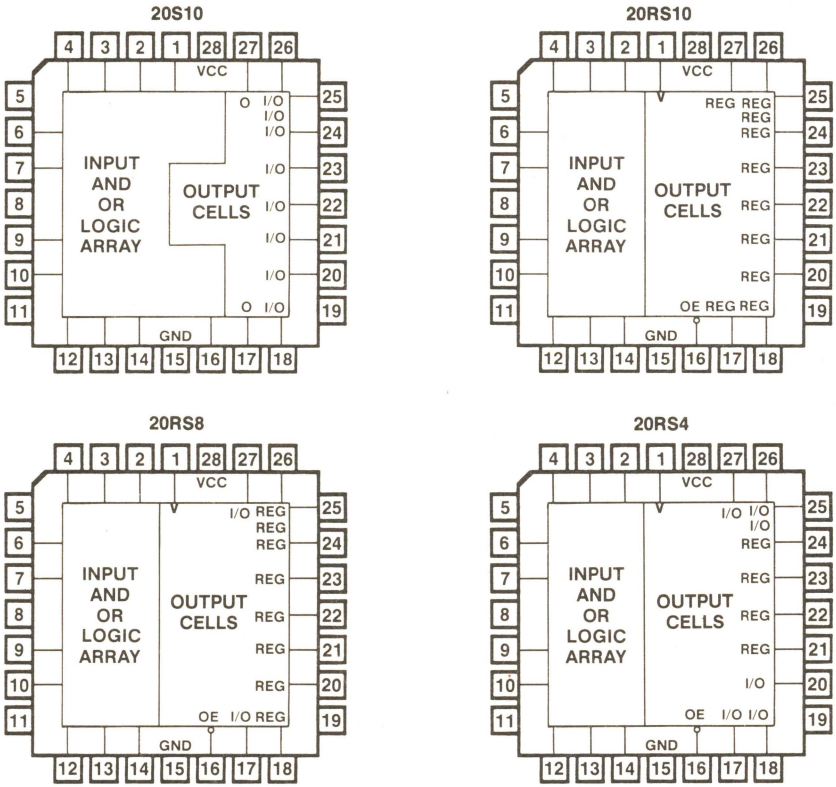
## Packages

The commercial PAL20RS10 Series is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), plastic leaded chip carrier (NL), and small outline (SG) packages.

## DIP/SO Pinouts




## PLCC Pinouts




5

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 15 | 10 | | ns |
| | | High | 15 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 20RS10, 20RS8, 20RS4 | 35 | 25 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 175 | 240 | mA |

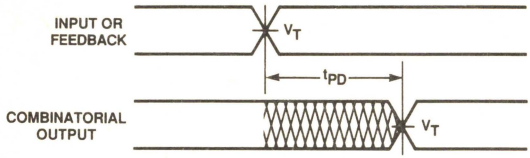
1. These are absolute voltages with respect to the ground pin on the device andi nclude all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

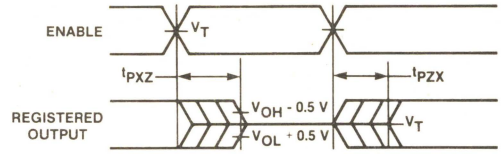
| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 20S10, 20RS8, 20RS4 | | Polarity fuse intact | $R_1$ = 200 Ω<br>$R_2$ = 390 KΩ | | 25 | 35 | ns |
| | | | Polarity fuse programmed | | | 30 | 40 | |
| $t_{CLK}$ | Clock to output or feedback | | | | | 12 | 17 | ns |
| $t_{CF}$ | Clock to feedback | | | | | 10 | 15 | ns |
| $t_{PZX}$ | Pin 13 to output enable except 20S10 | | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 13 to output disable except 20S10 | | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 20S10, 20RS8, 20RS4 | | | 25 | 35 | ns |
| $t_{ER}$ | Input to output disable | | 20S10, 20RS8, 20RS4 | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External | 20RS10, 20RS8, 20RS4 | | 19 | 27 | | MHz |
| | | Internal | | | 20 | 28 | | |
| | | No feedback | | | 33 | 50 | | |

5

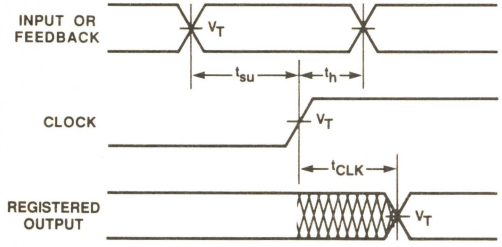
## Switching Waveforms



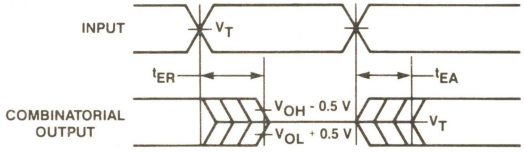

Combinatorial Output



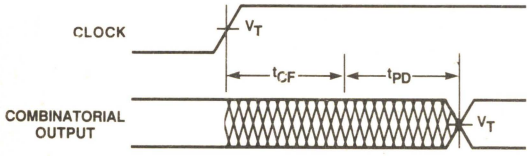

Pin 13 to Output Disable/Enable



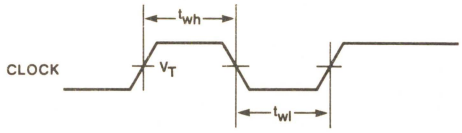

Registered Output



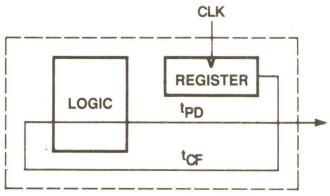

Input to Output Disable/Enable




Clock to Feedback to Combinatorial Output (See Path Below)

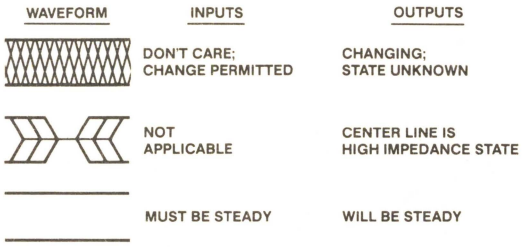



Clock Width




Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagrams




## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Power-Up Reset Waveform

(refer to page 5-164)

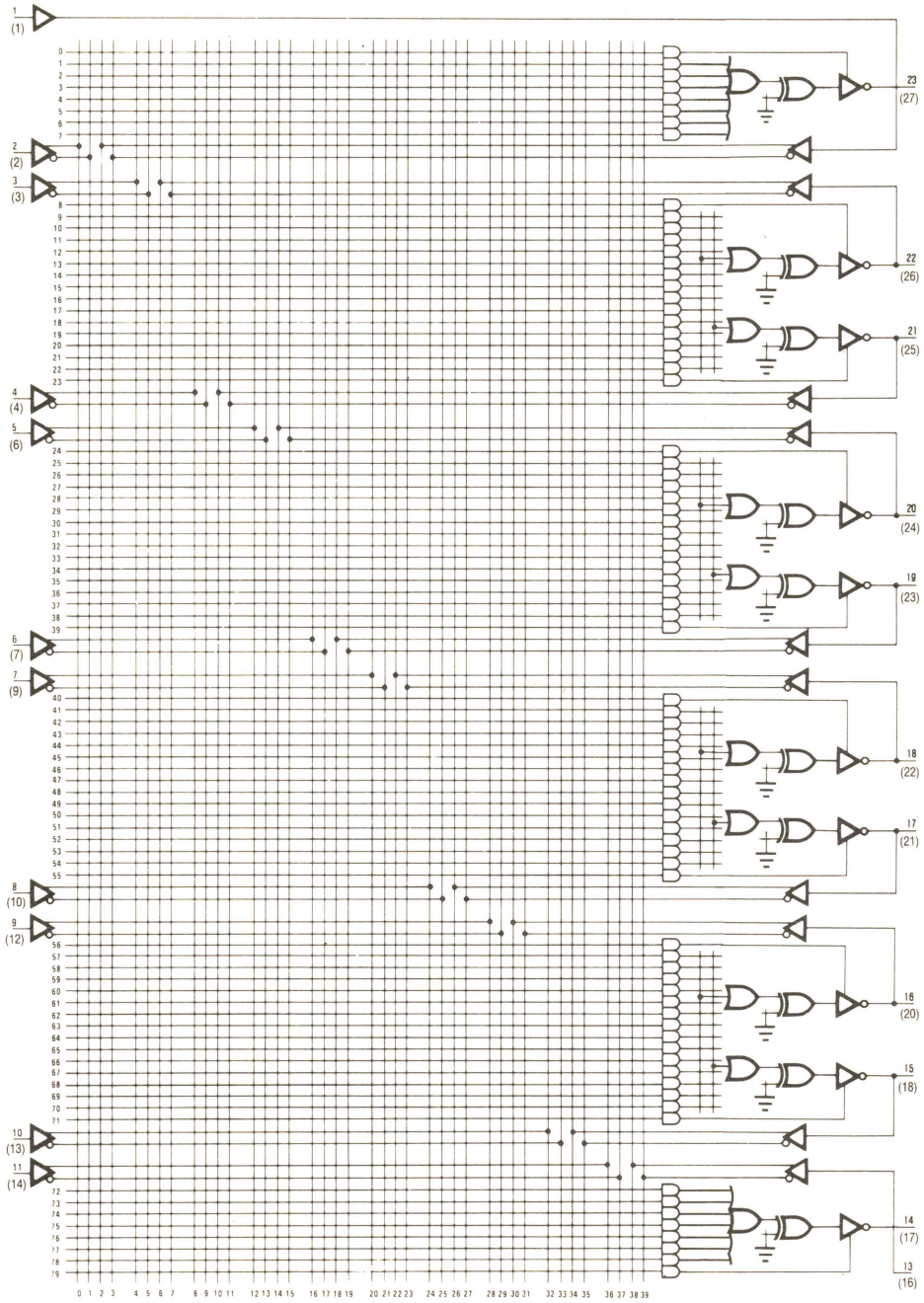
## Register Preload Waveform

(refer to page 5-164)

## Schematic of Inputs and Outputs

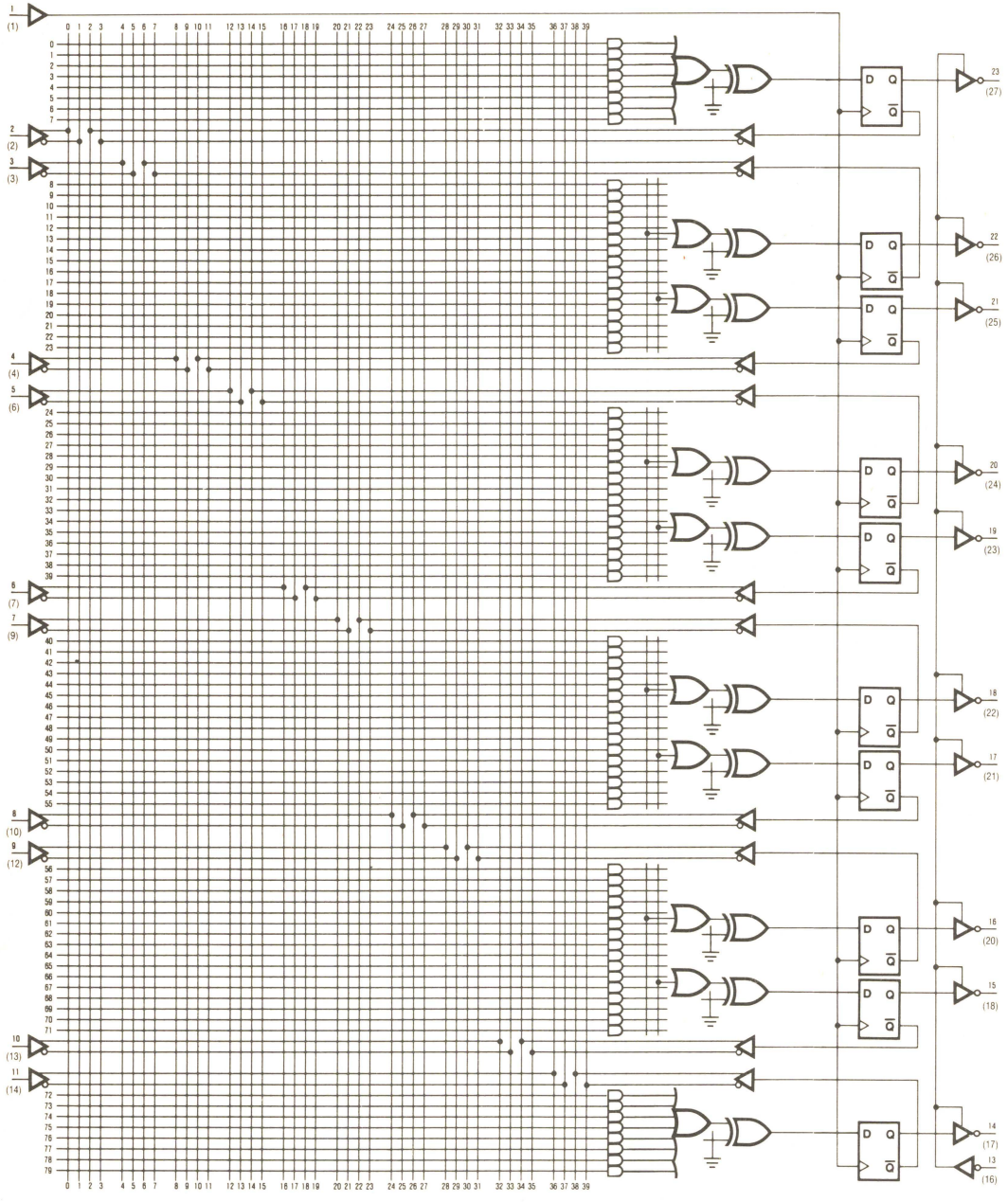
(refer to page 5-164)

## 20S10

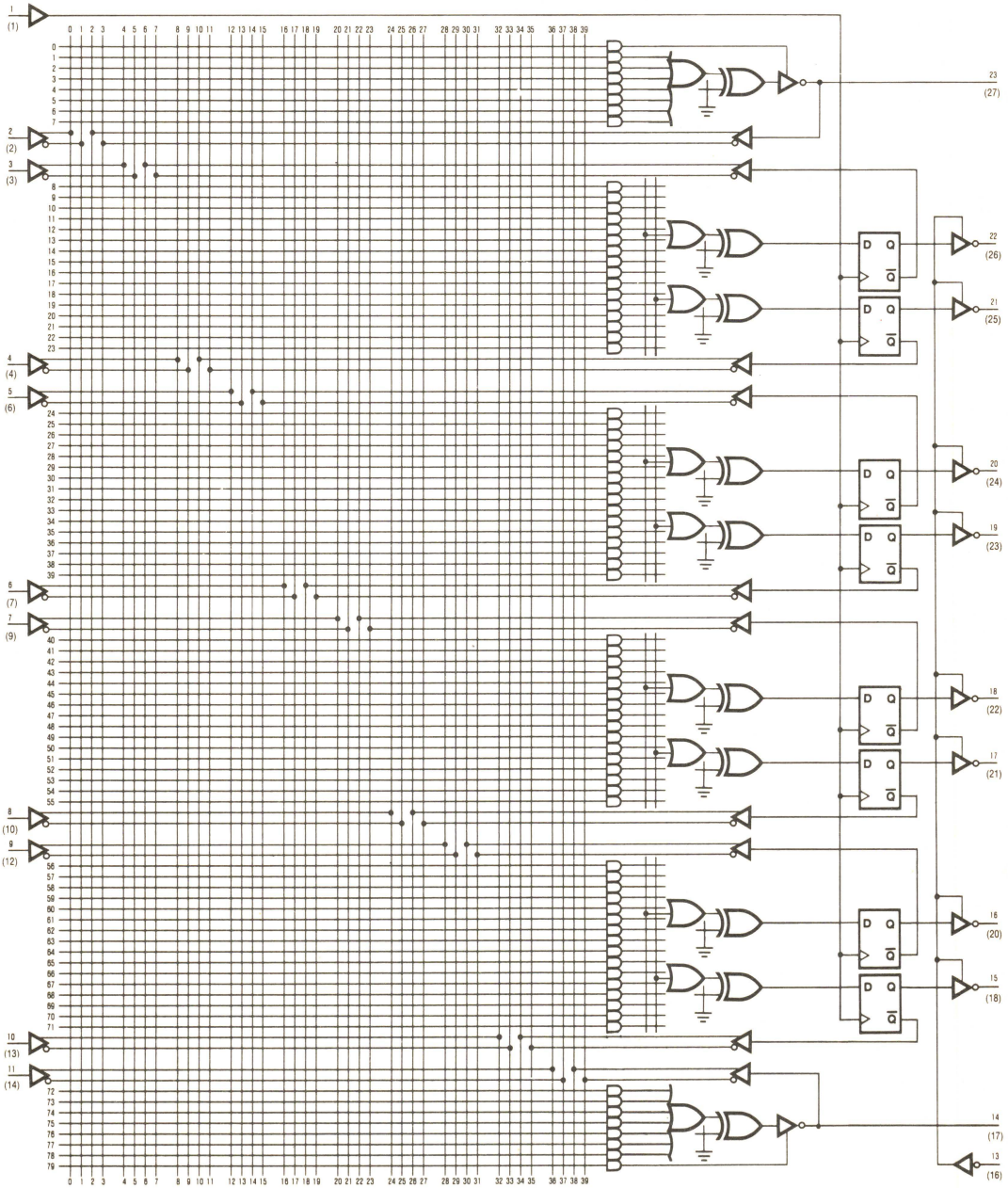

1 (1)
2 (2)
3 (3)
4 (4)
5 (6)
6 (7)
7 (9)
8 (10)
9 (12)
10 (13)
11 (14)
23 (27)
22 (26)
21 (25)
20 (24)
19 (23)
18 (22)
17 (21)
16 (20)
15 (18)
14 (17)
13 (16)
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

5

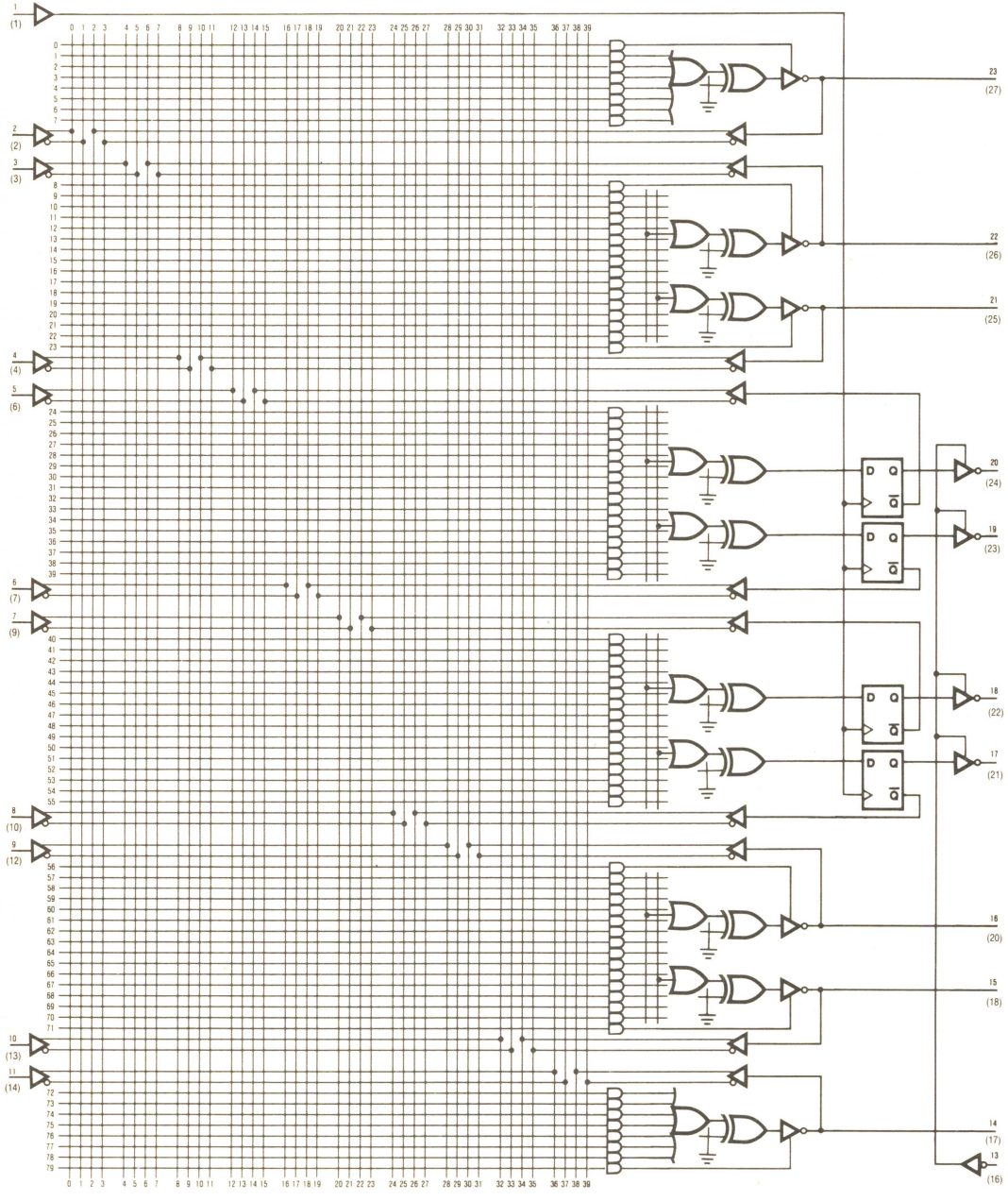
## 20RS10



## 20RS8



## 20RS4



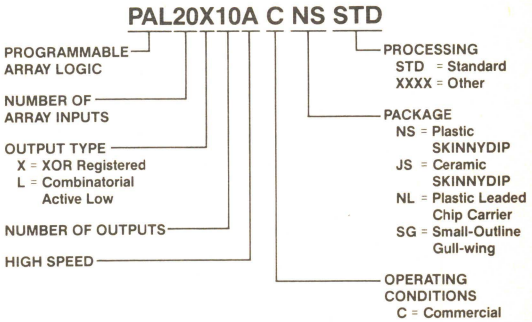
# PAL20X10A Series

# 20L10A, 20X10A 20X8A, 20X4A

## Features/Benefits

- XOR gates on registered outputs
- Efficient implementation of counters
- Register preload
- Power-up reset
- Security fuse

## Ordering Information




## PAL20X10A Series

| | ARRAY INPUTS | OUTPUTS | | $t_{PD}$* (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|---|
| | | COMBINATORIAL | REGISTERED | | |
| PAL20L10A | 20 | 10 | 0 | 30 | 165 |
| PAL20X10A | 20 | 0 | 10 | 30 | 180 |
| PAL20X8A | 20 | 2 | 8 | 30 | 180 |
| PAL20X4A | 20 | 6 | 4 | 30 | 180 |

## Description

The PAL20X10A Series offers Exclusive-OR (XOR) gates preceding each register. The XOR gate has as its inputs two sums, each of two product terms. The XOR gate is very efficient for counting applications.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

The register devices have ten dedicated input lines, and each combinatorial output is an I/O pin. The combinatorial device has twelve dedicated input lines, and only eight of the ten combinatorial outputs are I/O pins. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Polarity

All outputs are active low.

## Preload and Power-Up Reset

The 20X10A Series offers register preload for device testability. The registers can be preloaded from the outputs by using supervoltages in order to simplify functional testing. The 20X10A Series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

## Packages

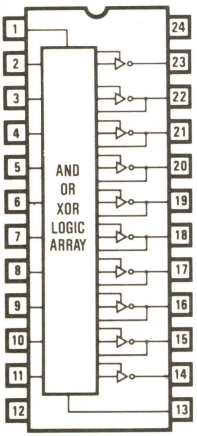
The commercial PAL20X10A Series is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), plastic leaded chip carrier (NL), and small outline (SG) packages.



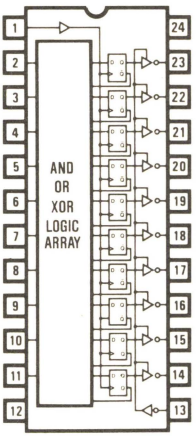
10303A
JANUARY 1988

## DIP/SO Pinouts

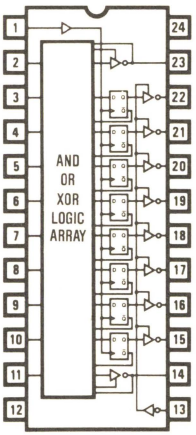
20L10A



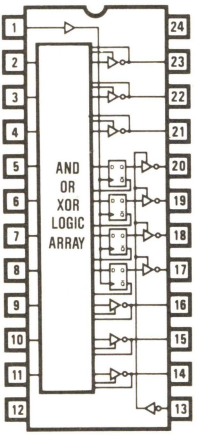

20X10A




20X8A

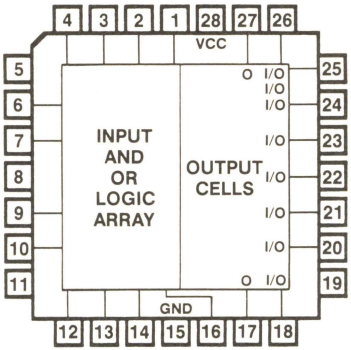



20X4A

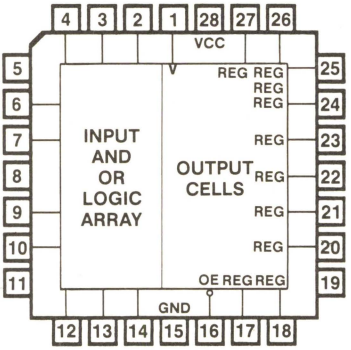



## PLCC Pinouts

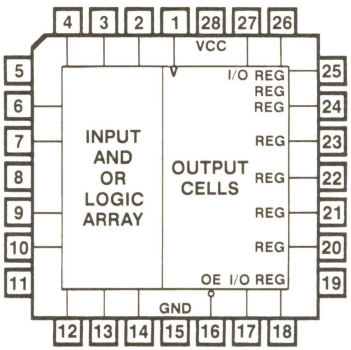
20L10A



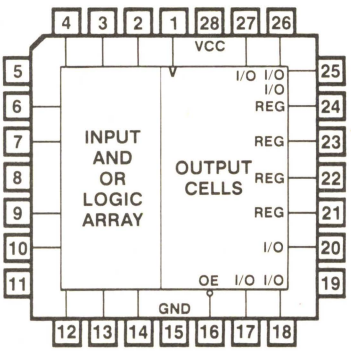

20X10A




20X8A




20X4A



## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 25 | 15 | | ns |
| | | High | 15 | 7 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 20X10A, 20X8A, 20X4A | 30 | 20 | | ns |
| $t_h$ | Hold time | | 0 | -15 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

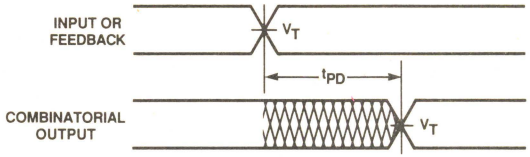
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | 20X10A,20X8A,20X4A | | 140 | 180 | mA |
| | | | 20L10A | | 115 | 165 | |

1. The PAL20X10A Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

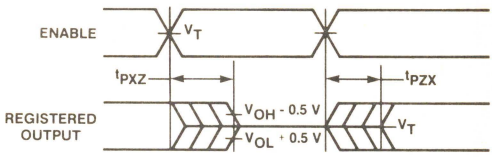
## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 20L10A, 20X8A, 20X4A | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 23 | 30 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 10 | 15 | ns |
| $t_{PZX}$ | Pin 13 to output enable except 20L10A | | | | | 11 | 20 | ns |
| $t_{PXZ}$ | Pin 13 to output disable except 20L10A | | | | | 10 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 20L10A, 20X8A, 20X4A | | | 19 | 30 | ns |
| $t_{ER}$ | Input to output disable | | 20L10A, 20X8A, 20X4A | | | 15 | 30 | ns |
| $f_{MAX}$ | Maximum frequency | External | 20L10A, 20X8A, 20X4A | | 22.2 | 32 | | MHz |
| | | No feedback | | | 25 | 45 | | |

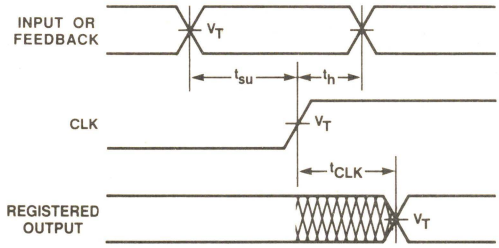
## Switching Waveforms



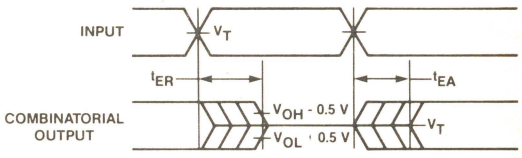

Combinatorial Output



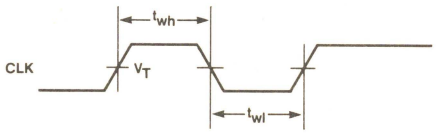

Pin 13 to Output Disable/Enable




Registered Output

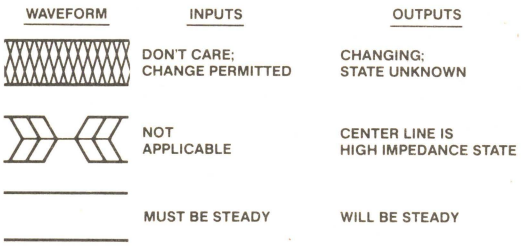



Input to Output Disable/Enable




Clock Width

## Key to Timing Diagrams




**Switching Test Load**
(refer to page 5-164)

**Programmers/Development Systems**
(refer to Programmer Reference Guide, page 3-81)

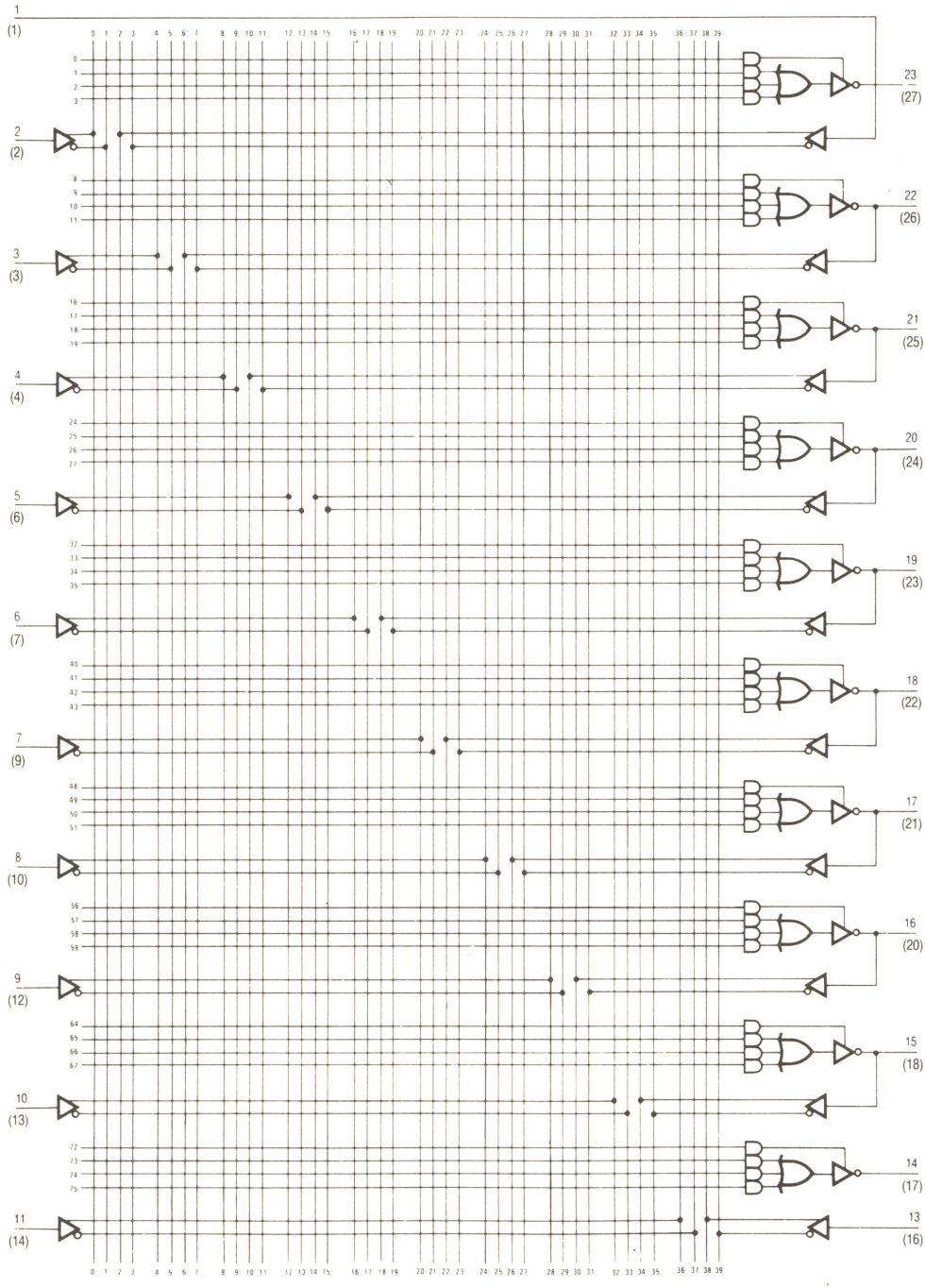
**Register Preload Waveform**
(refer to page 5-164)

**Power-Up Reset Waveform**
(refer to page 5-164)

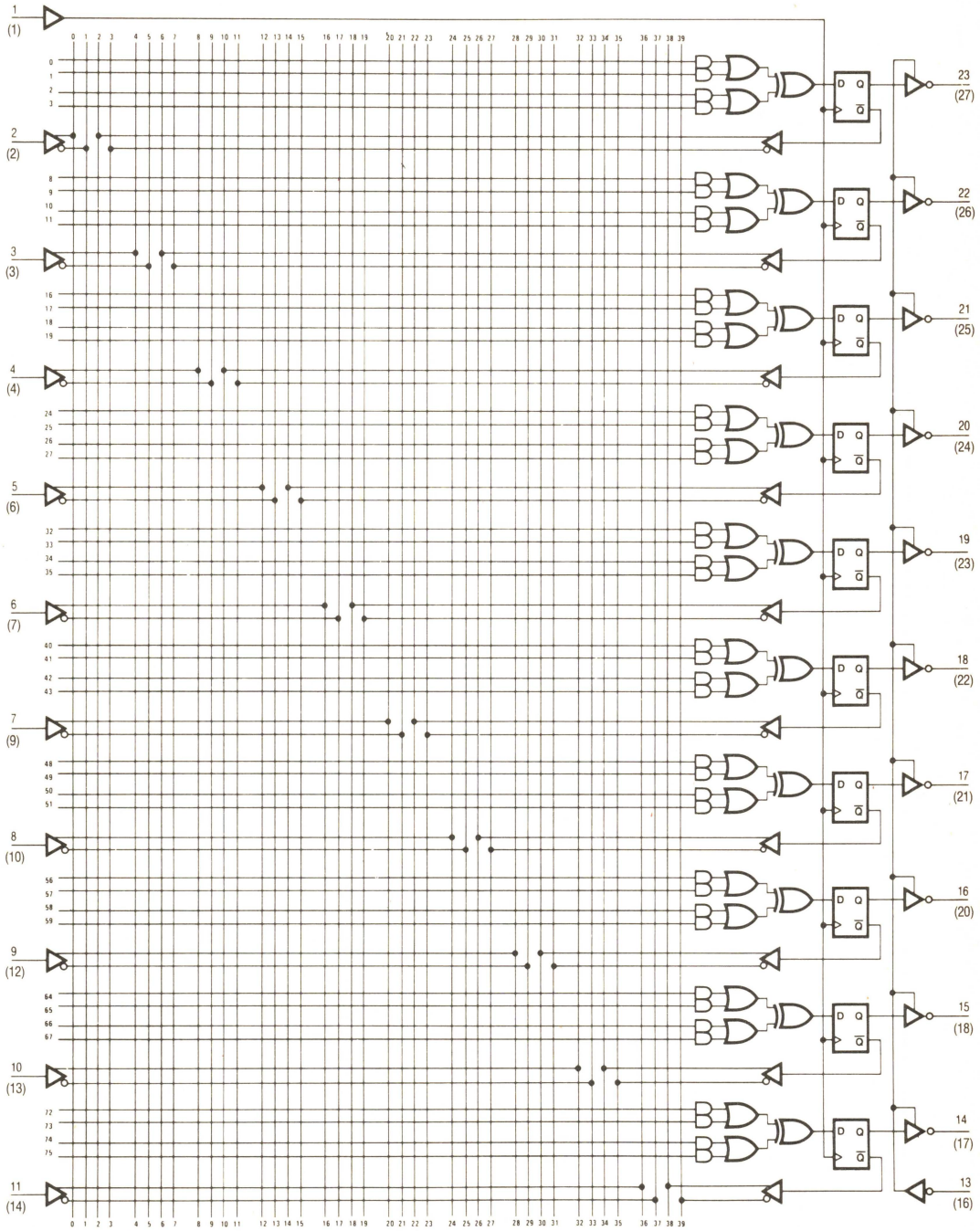
**Schematic of Inputs and Outputs**
(refer to page 5-164)

**Package Drawings**
(refer to PAL Device Package Outlines, page 3-179)

## 20L10

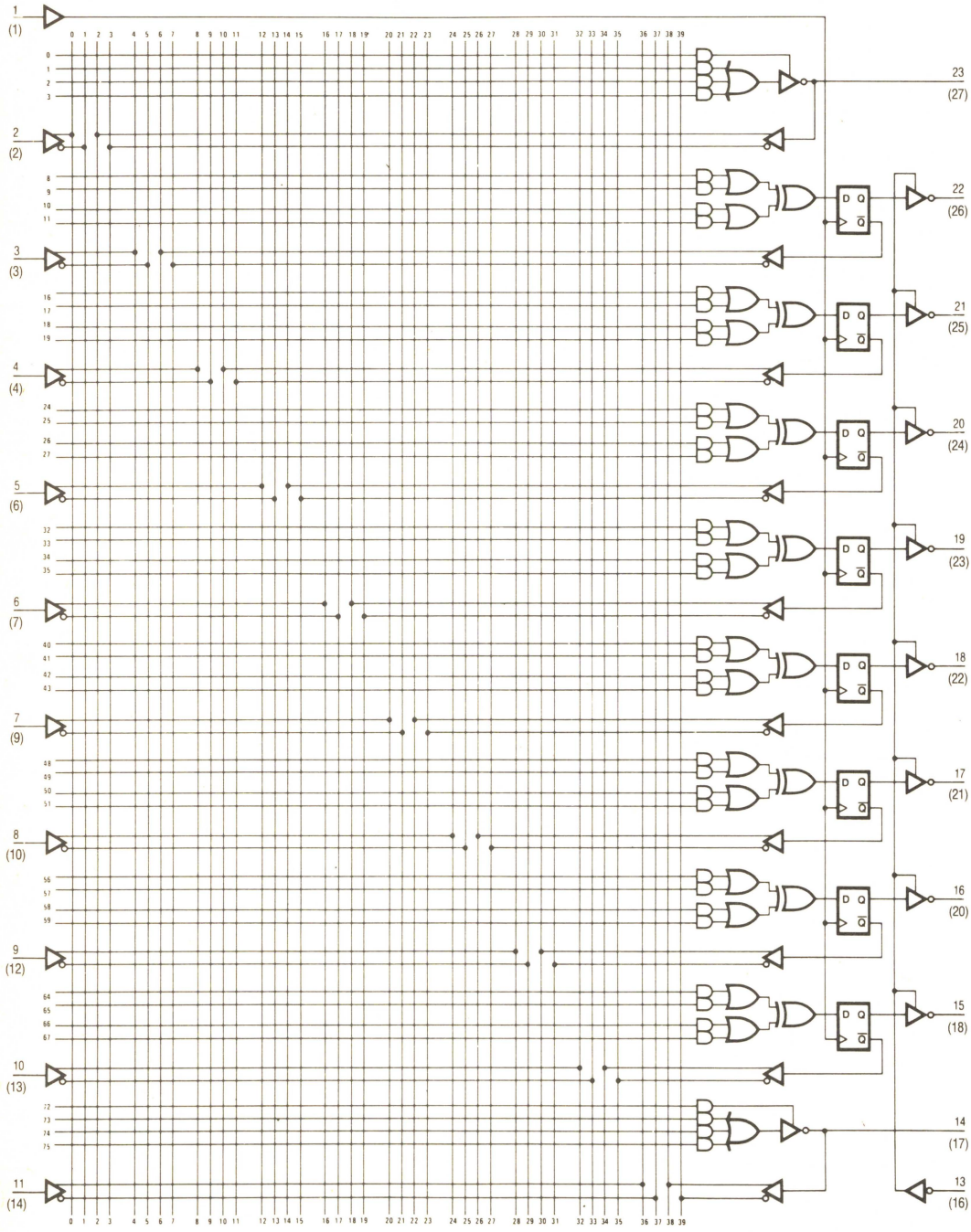

1
(1)
2
(2)
3
(3)
4
(4)
5
(6)
6
(7)
7
(9)
8
(10)
9
(12)
10
(13)
11
(14)
23
(27)
22
(26)
21
(25)
20
(24)
19
(23)
18
(22)
17
(21)
16
(20)
15
(18)
14
(17)
13
(16)

## 20X10

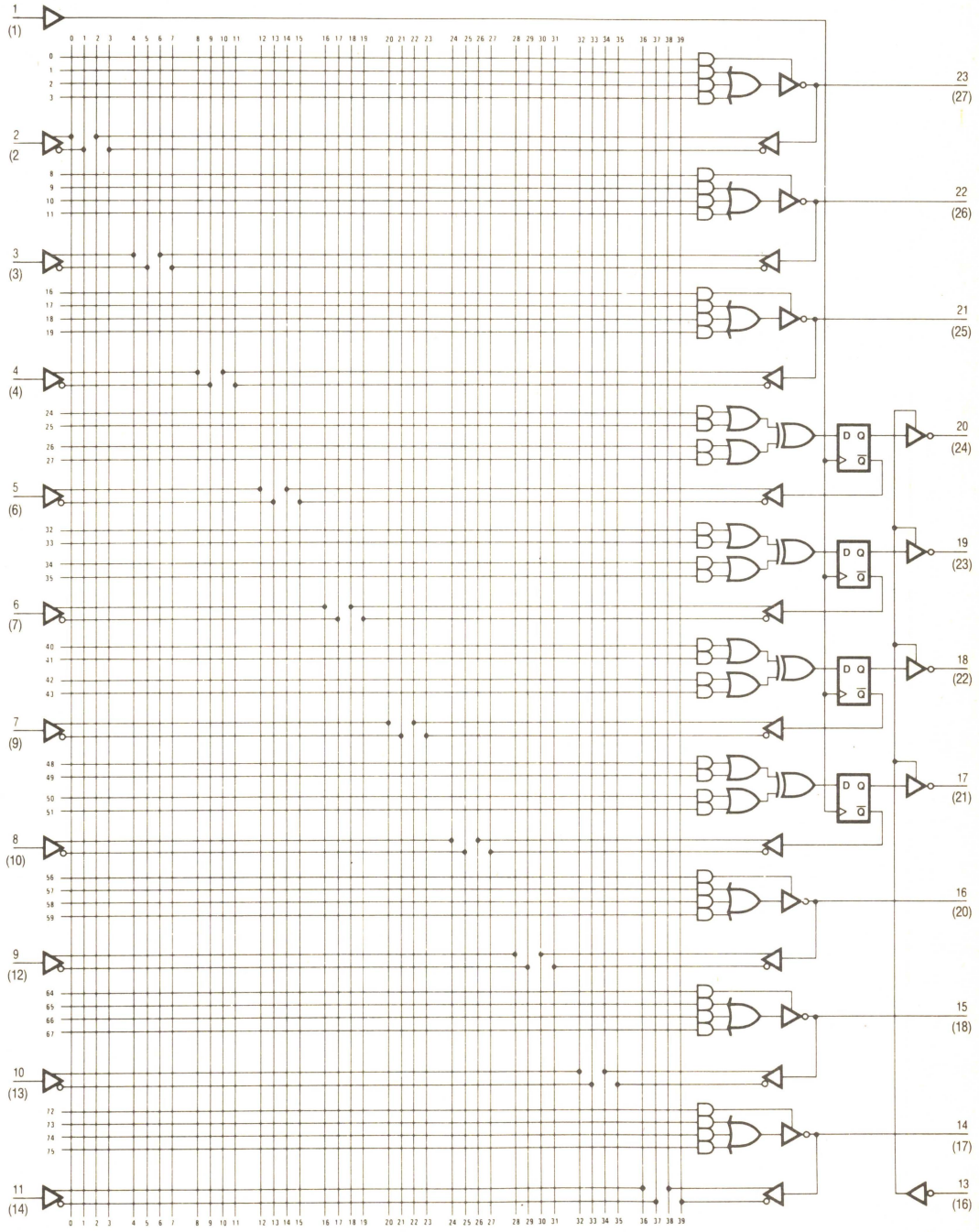

1
(1)
2
(2)
3
(3)
4
(4)
5
(6)
6
(7)
7
(9)
8
(10)
9
(12)
10
(13)
11
(14)
23
(27)
22
(26)
21
(25)
20
(24)
19
(23)
18
(22)
17
(21)
16
(20)
15
(18)
14
(17)
13
(16)
D Q
Q

5

## 20X8


1 (1)
2 (2)
3 (3)
4 (4)
5 (6)
6 (7)
7 (9)
8 (10)
9 (12)
10 (13)
11 (14)
23 (27)
22 (26)
21 (25)
20 (24)
19 (23)
18 (22)
17 (21)
16 (20)
15 (18)
14 (17)
13 (16)

## 20X4


1
(1)
2
(2
3
(3)
4
(4)
5
(6)
6
(7)
7
(9)
8
(10)
9
(12)
10
(13)
11
(14)
23
(27)
22
(26)
21
(25)
20
(24)
19
(23)
18
(22)
17
(21)
16
(20)
15
(18)
14
(17)
13
(16)

# PAL20R8 Family

# 20L8, 20R8
# 20R6, 20R4

## Features/Benefits

- **Standard 24-pin architectures**
- **TTL and CMOS versions**
- **High speed, as fast as 15 ns tPD for PAL20R8B Series**
- **Low power, as low as zero standby for PALC20R8Z Series**
- **Security fuse/cell on all devices**

## Description

The PAL20R8 Series consists of four devices, each with twenty array inputs and eight outputs. The devices have either 0, 4, 6, or 8 registered outputs, with the remaining being combinatorial.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

The registered devices have twelve dedicated input lines, and each combinatorial output is an I/O pin. The combinatorial device has fourteen dedicated input lines, and only six of the eight combinatorial outputs are I/O pins. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. On combinatorial outputs, a product term controls the buffer, allowing enable and disable to be a function of any combination of device inputs or output feedback. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Polarity

All outputs are active low.

## Performance

Several speed/power versions are available.

| SUFFIX | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|
| B | 15 | 210 |
| B-2 | 25 | 105 |
| A | 25 | 210 |
| A-2 | 35 | 105 |
| Z-35 | 35 | 0.1 |
| Z-45 | 45 | 0.1 |

## Preload and Power-Up Reset

The B-2 and CMOS Series offer register preload for device testability. The registers can be preloaded from the outputs by using super-voltages (see waveforms at end of section) in order to simplify functional testing. The B-2 Series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

## Packages

The commercial PAL20R8 Series is available in the plastic SKINNYDIP (NS) and ceramic SKINNYDIP (JS) packages. The PAL20R8B/A/A-2 Series is available in the plastic leaded chip carrier with no-connects on 4, 8, 11, and 19 (NL), while the PAL20R8B-2/Z-35/Z-45 Series is available in the plastic leaded chip carrier with no-connects on 1, 8, 15, and 22 (FN). The PALC20R8Z-35/45 Series is also available in the ceramic windowed SKINNYDIP (QS) package.

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

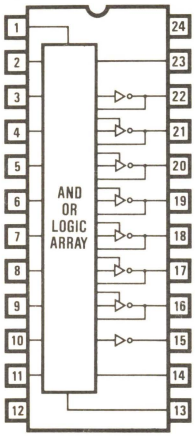
## PAL20R8 Series

| DEVICE | DEDICATED INPUTS | OUTPUTS | |
|---|---|---|---|
| | | COMBINATORIAL | REGISTERED |
| PAL20L8 | 12 | 8 (6 I/O) | 0 |
| PAL20R8 | 10 | 0 | 8 |
| PAL20R6 | 10 | 2 I/O | 6 |
| PAL20R4 | 10 | 4 I/O | 4 |

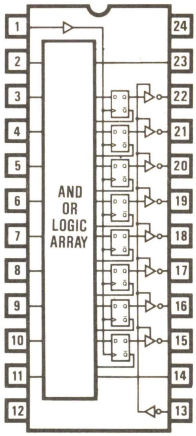
10294A
JANUARY 1988

## DIP Pinouts

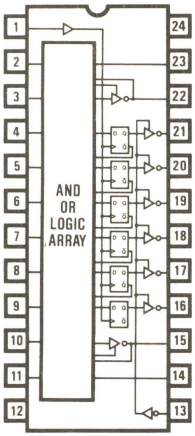
20L8A/A-2/B/B-2/Z-35/Z-45



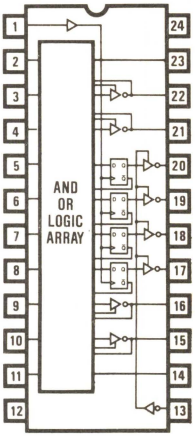

20R8A/A-2/B/B-2/Z-35/Z-45




20R6A/A-2/B/B-2/Z-35/Z-45

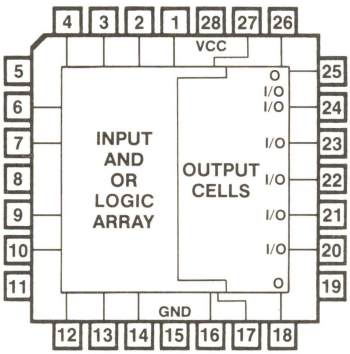



20R4A/A-2/B/B-2/Z-35/Z-45

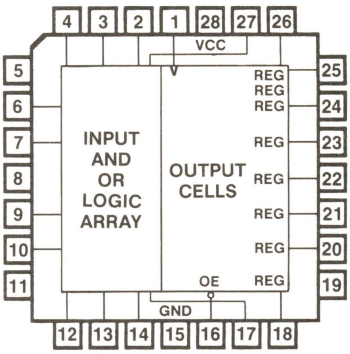



## PLCC Pinouts (NL)

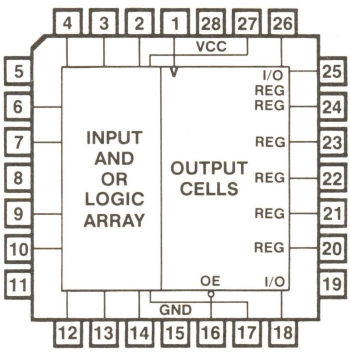
20L8A/A-2/B



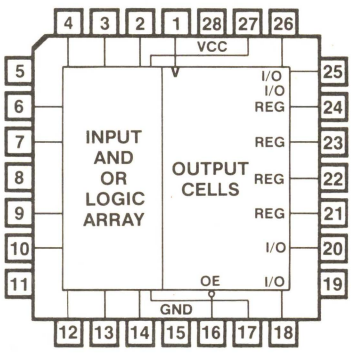

20R8A/A-2/B




20R6A/A-2/B



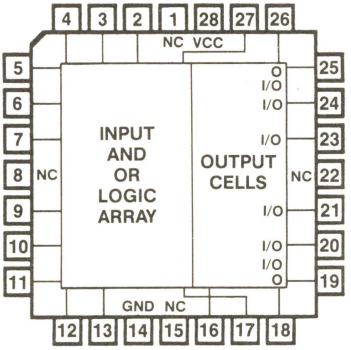

20R4A/A-2/B



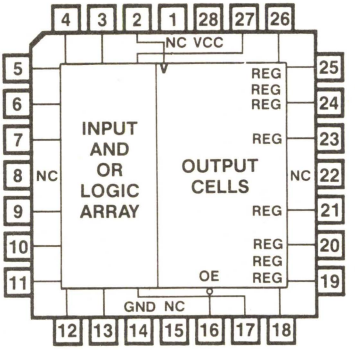

5

## PLCC Pinouts (FN)

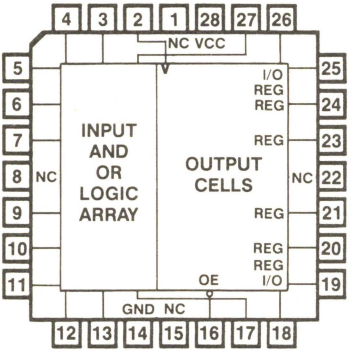
PAL20L8B-2/Z-35/Z-45



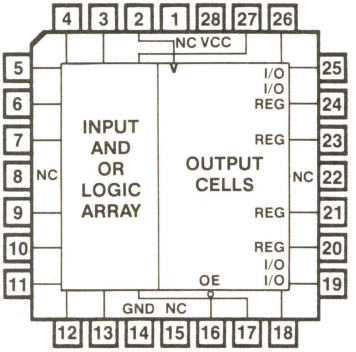

PAL20R8B-2/Z-35/Z-45




PAL20R6B-2/Z-35/Z-45




PAL20R4B-2/Z-35/Z-45




## Ordering Information — Newer Products

PALC20R8Z-35 C QS STD

- PROGRAMMABLE ARRAY LOGIC
- TECHNOLOGY: None = Bipolar; C = CMOS
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE: R = Registered; L = Active Low Combinatorial
- NUMBER OF OUTPUTS
- POWER: None = Standard; Z = Zero Standby Power
- SPEED (tPD)
- OPERATING CONDITIONS: C = Commercial; I = Industrial (CMOS Only)
- PACKAGE: NS = Plastic SKINNYDIP; JS = Ceramic SKINNYDIP; QS = Windowed SKINNYDIP (CMOS Only); FN = Plastic Leaded Chip Carrier; SG = Small-Outline Gull-wing
- PROCESSING: STD = Standard; XXXX = Other

## Ordering Information — Older Products

PAL20R8B-2 C NS STD

- PROGRAMMABLE ARRAY LOGIC
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE: R = Registered; L = Active Low Combinatorial
- NUMBER OF OUTPUTS
- SPEED: A = High Speed; B = Very High Speed
- POWER: None = Standard; -2 = Half Power
- OPERATING CONDITIONS: C = Commercial
- PACKAGE: NS = Plastic DIP; JS = Ceramic DIP; NL = Plastic Leaded Chip Carrier; FN = Plastic Leaded Chip Carrier (with Center No-connects) (PAL20R8B-2 Series Only); SG = Small-Outline Gull-wing
- PROCESSING: STD = Standard; XXXX = Other

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 10 | 6 | | ns |
| | | High | 12 | 8 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 20R8B, 20R6B, 20R4B | 15 | 10 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 140 | 210 | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 20L8B, 20R6B, 20R4B | $R_1$ = 200 Ω, $R_2$ = 390 Ω | | 12 | 15 | ns |
| $t_{CLK}$ | Clock to output or feedback except 20L8B | | | | | 8 | 12 | ns |
| $t_{PZX}$ | Pin 13 to output enable except 20L8B | | | | | 10 | 15 | ns |
| $t_{PXZ}$ | Pin 13 to output disable except 20L8B | | | | | 8 | 12 | ns |
| $t_{EA}$ | Input to output enable | | 20L8B, 20R6B, 20R4B | | | 12 | 18 | ns |
| $t_{ER}$ | Input to output disable | | 20L8B, 20R6B, 20R4B | | | 12 | 15 | ns |
| $f_{MAX}$ | Maximum frequency | External | 20R8B, 20R6B, 20R4B | | 37 | 40 | | MHz |
| | | No feedback | | | 45 | 50 | | |

1. The PAL20R8B Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 20R8B-2, 20R6B-2, 20R4B-2 | 15 | 10 | | ns |
| | | High | | 15 | 10 | | |
| $t_{su}$ | Setup time from input or feedback to clock | | | 25 | 15 | | ns |
| $t_h$ | Hold time | | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 3.4 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 80 | 105 | mA |
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 2.0 V at f = 1 MHz | | | 6 | | pF |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | 9 | | pF |

Notes: 1. The PAL20R8B-2 Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 20L8B-2, 20R6B-2, 20R4B-2 | | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 15 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback | 20R8B-2, 20R6B-2, 20R4B-2 | | | 10 | 15 | ns |
| $t_{PZX}$ | Pin 13 to output enable | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 13 to output disable | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | 20L8B-2, 20R6B-2, 20R4B-2 | | | 10 | 25 | ns |
| $t_{ER}$ | Input to output disable | | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency 20R8B-2, 20R6B-2, 20R4B-2 | External | | 25 | 30 | | MHz |
| | | Internal | | 28.5 | 35 | | |
| | | No feedback | | 33.3 | 40 | | |

5

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | | -65° C to +150° C |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 15 | 7 | | ns |
| | | High | 15 | 7 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 20R8A, 20R6A, 20R4A | 25 | 15 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -90 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 160 | 210 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 20L8A, 20R6A, 20R4A | $R_1$ = 200 Ω<br>$R_2$ = 390 Ω | | 15 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | | | 10 | 15 | ns |
| $t_{CF}$ | Clock to feedback | | | | | 8 | 10 | ns |
| $t_{PZX}$ | Pin 13 to output enable except 20L8A | | | | | 10 | 20 | ns |
| $t_{PXZ}$ | Pin 13 to output disable except 20L8A | | | | | 11 | 20 | ns |
| $t_{EA}$ | Input to output enable | | 20L8A, 20R6A, 20R4A | | | 10 | 25 | ns |
| $t_{ER}$ | Input to output disable | | 20L8A, 20R6A, 20R4A | | | 13 | 25 | ns |
| $f_{MAX}$ | Maximum frequency | External | 20R8A, 20R6A, 20R4A | | 25 | 40 | | MHz |
| | | Internal | | | 28.5 | 43 | | |
| | | No feedback | | | 33 | 71 | | |

5

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | −0.5 V to 7.0 V | −0.5 V to 12.0 V |
| Input voltage | −1.5 V to 5.5 V | −1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | −65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 25 | 10 | | ns |
| | | High | 25 | 10 | | |
| $t_{su}$ | Set up time from input or feedback to clock | 20R8A-2, 20R6A-2, 20R4A-2 | 35 | 25 | | ns |
| $t_h$ | Hold time | | 0 | −15 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = −18 mA | | −0.8 | −1.5 | V |
| $I_{IL}$[3] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | −0.02 | −0.25 | mA |
| $I_{IH}$[3] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 24 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = −3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | −100 | μA |
| $I_{OZH}$[3] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[4] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | −30 | −70 | −130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 80 | 105 | mA |

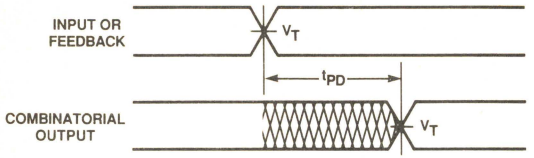
1. The PAL20R8A-2 Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
4. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

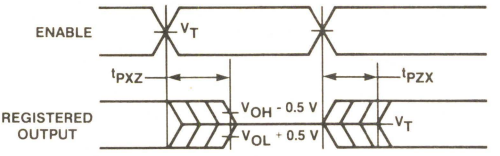
| SYMBOL | PARAMETER | | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | 20L8A-2, 20R6A-2, 20R4A-2 | Commercial $R_1 = 200\ \Omega$ $R_2 = 390\ \Omega$ | | 25 | 35 | ns |
| $t_{CLK}$ | Clock to output or feedback except 20L8A-2 | | | | | 15 | 25 | ns |
| $t_{PZX}$ | Pin 13 to output enable except 20L8A-2 | | | | | 15 | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable except 20L8A-2 | | | | | 15 | 25 | ns |
| $t_{EA}$ | Input to output enable | | 20L8A-2, 20R6A-2, 20R4A-2 | | | 25 | 35 | ns |
| $t_{ER}$ | Input to output disable | | 20L8A-2, 20R6A-2, 20R4A-2 | | | 25 | 35 | ns |
| $f_{MAX}$ | Maximum frequency | External | 20R8A-2, 20R6A-2, 20R4A-2 | | 16 | 25 | | MHz |
| | | No feedback | | | 20 | 50 | | |

5

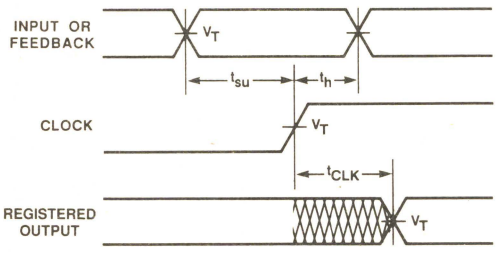
## Switching Waveforms



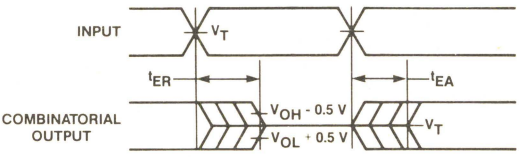

Combinatorial Output



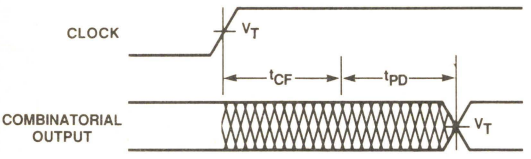

Pin 13 to Output Disable/Enable



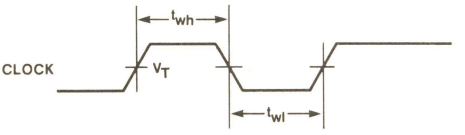

Registered Output



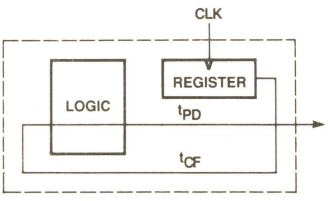

Input to Output Disable/Enable




Clock to Feedback to Combinatorial Output (See Path Below)

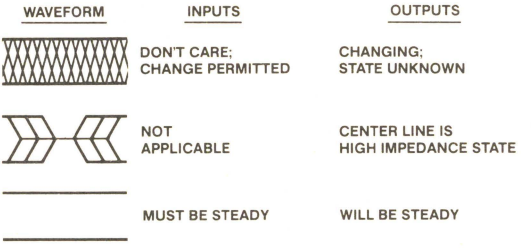



Clock Width




Notes:
1. VT = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagrams




## Switching Test Load
(refer to page 5-164)

## Programmers/Development Systems
(refer to Programmer Reference Guide, page 3-81)

## Register Preload Waveform
(refer to page 5-164)

## Power-Up Reset Waveform
(refer to page 5-164)

## Schematic of Inputs and Outputs
(refer to page 5-164)

## Absolute Maximum Ratings

Supply voltage $V_{CC}$ ..... -0.5 V to 7.0 V
DC input voltage, $V_I$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output voltage, $V_O$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output source/sink current per output pin, $I_O$ ..... ±35 mA
DC $V_{CC}$ or ground current, $I_{CC}$ or $I_{GND}$ ..... ±100 mA
Input diode current, $I_{IK}$:
$V_I$<0 ..... -20 mA
$V_I$>$V_{CC}$ ..... +20 mA
Output diode current, $I_{OK}$:
$V_O$<0 ..... -20 mA
$V_O$>$V_{CC}$ ..... +20 mA
Storage temperature ..... -65°C to 150°C
Static discharge voltage ..... >2001 V
Latchup current ..... >100 mA

## Operating Conditions

| SYMBOL | PARAMETER | | INDUSTRIAL[1] | | | | | | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | -50 | | | -40 | | | -45 | | | -35 | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | MIN | TYP | MAX | MIN | TYP | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | 20R8 20R6 20R4 | 15 | 10 | | 15 | 10 | | 15 | 10 | | 15 | 10 | | ns |
| $t_{su}$ | Setup time from input or feedback to clock | | 45 | 30 | | 35 | 25 | | 40 | 30 | | 30 | 25 | | ns |
| $t_h$ | Hold time | | 0 | -15 | | 0 | -15 | | 0 | -15 | | 0 | -15 | | ns |
| $T_A$ | Operating free-air temperature | | -40 | 25 | 85 | -40 | 25 | 85 | 0 | 25 | 75 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | 0 | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = GND | | | -1 | µA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = $V_{CC}$ | | | 1 | µA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.25 | 0.45 | V |
| | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 µA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -6 mA[3] | 3.76 | 4.1 | | V |
| | | $V_{CC}$ = 5 V | $I_{OH}$ = -1 µA | 4.95 | | | |
| $I_{OZL}$ | Off-state output current | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | -10 | µA |
| $I_{OZH}$ | | | $V_O$ = $V_{CC}$ | | 0 | 10 | µA |
| $I_{CC}$ | Standby supply current[4] | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | µA |
| | Operating supply current[5] | f = 1 MHz | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | 7 | 10 | mA |
| $C_{IN}$ | Input capacitance[8] | $V_{IN}$ = 2.0 V at f = 1 MHz | | | 6 | | pF |
| $C_{OUT}$ | Output capacitance[8] | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | 9 | | pF |

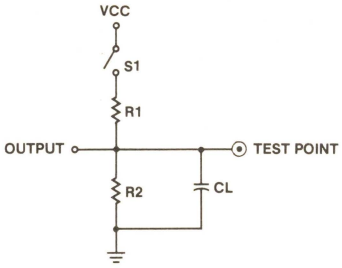
5

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER[7] | | | INDUSTRIAL | | | | | | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | -50 | | | -40 | | | -45 | | | -35 | | | |
| | | | | MIN | TYP | MAX | MIN | TYP | MAX | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{PD}$ | Input or feedback to output | | | | 45 | 50 | | 35 | 40 | | 40 | 45 | | 30 | 35 | ns |
| | 20L8, 20R6, 20R4 | | | | | | | | | | | | | | | |
| $t_{CLK}$ | Clock to output or feedback | | 20R8<br>20R6<br>20R4 | | 20 | 25 | | 15 | 20 | | 20 | 25 | | 15 | 20 | ns |
| $t_{PZX}$ | Pin 13 (DIP) to output enable | | 20L8<br>20R6<br>20R4 | | 20 | 25 | | 15 | 20 | | 20 | 25 | | 15 | 20 | ns |
| $t_{PXZ}$ | Pin 13 (DIP) to output disable | | | | | | | | | | | | | | | |
| $t_{EA}$[6] | Input to output enable | | 20R8<br>20R6<br>20R4 | | 45 | 50 | | 35 | 40 | | 40 | 45 | | 30 | 35 | ns |
| $t_{ER}$[6] | Input to output disable | | | | | | | | | | | | | | | |
| $f_{MAX}$ | Maximum frequency | External feedback ($1/t_{su}+t_{CLK}$) | | 14.2 | 20 | | 18.1 | 25 | | 15.3 | 20 | | 20 | 25 | | MHz |

1. The PALC20R8Z Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. JEDEC standard no. 7 for high-speed CMOS devices.
4. Disabled output pins = $V_{CC}$ or GND.
5. Frequency of any input or clock. See graph page 5.
6. Equivalent function to $t_{PZX}/t_{PXZ}$ but using product term control.
7. Test conditions (see Test Load) $R_1 = 440\ \Omega$, $R_2 = 190\ \Omega$.
8. Sampled but not 100% tested.

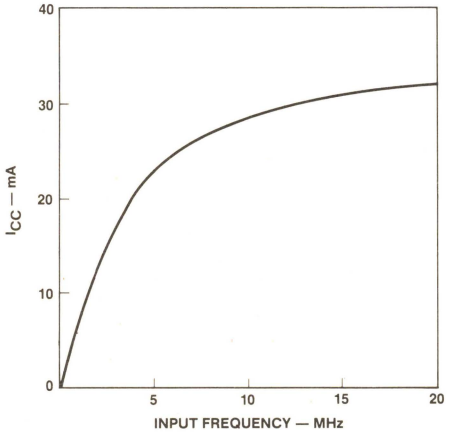
## Switching Test Load -PALC20R8Z Series




| SPECIFICATION | SWITCH S1 | $C_L$ | MEASURED OUTPUT VALUE |
|---|---|---|---|
| $t_{PD}$, $t_{CLK}$ | Closed | 50 pF | 1.5 V |
| $t_{PZX}$, $t_{EA}$ | Z->H: closed<br>Z->L: closed | 50 pF | 2.0 V<br>0.8 V |
| $t_{PXZ}$, $t_{ER}$ | H->Z: closed<br>L->Z: closed | 5 pF | H->Z: $V_{OH}$-0.5 V<br>L->Z: $V_{OL}$+0.5 V |

## $I_{CC}$ vs. Frequency–PALC20R8Z Series

Typical: $V_{CC}$ = 5 V, $T_A$ = 25°C

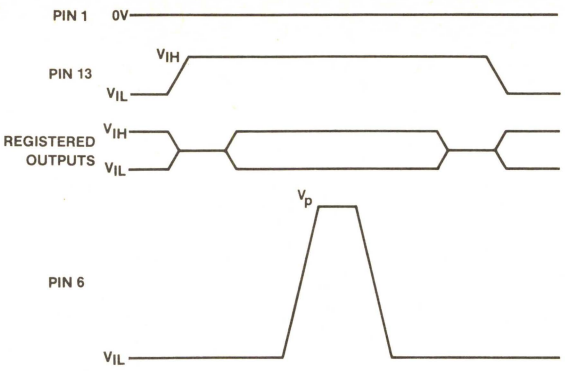



5

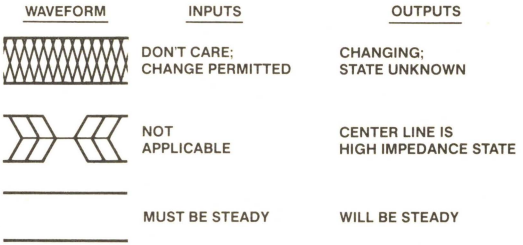
## Output Register PRELOAD –PALC20R8Z Series

The PRELOAD function allows the register to be loaded from data placed on the output pins. This feature aids functional testing of state sequencer designs by allowing direct setting of output states for improved test coverage. The PRELOAD procedure (using DIP pin numbers) is as follows:

1. Raise $V_{CC}$ to 5 V.
2. Disable output registers by setting pin 13 to $V_{IH}$. Set pin 1 to 0 V.
3. Apply $V_{IL}/V_{IH}$ (as desired) to all registered outputs.
4. Pulse pin 6 to $V_p$ (12 V), then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all registered outputs.
6. Lower pin 13 to $V_{IL}$ to enable the registered outputs.
7. Verify for $V_{OL}/V_{OH}$ at all registered outputs.




## Key to Timing Diagrams




## Programming and Erasing –PALC20R8Z Series

The PALC20R8Z Series can be programmed on standard logic programmers. The PALC20R8Z Series may be erased by ultraviolet light when contained in the windowed package.

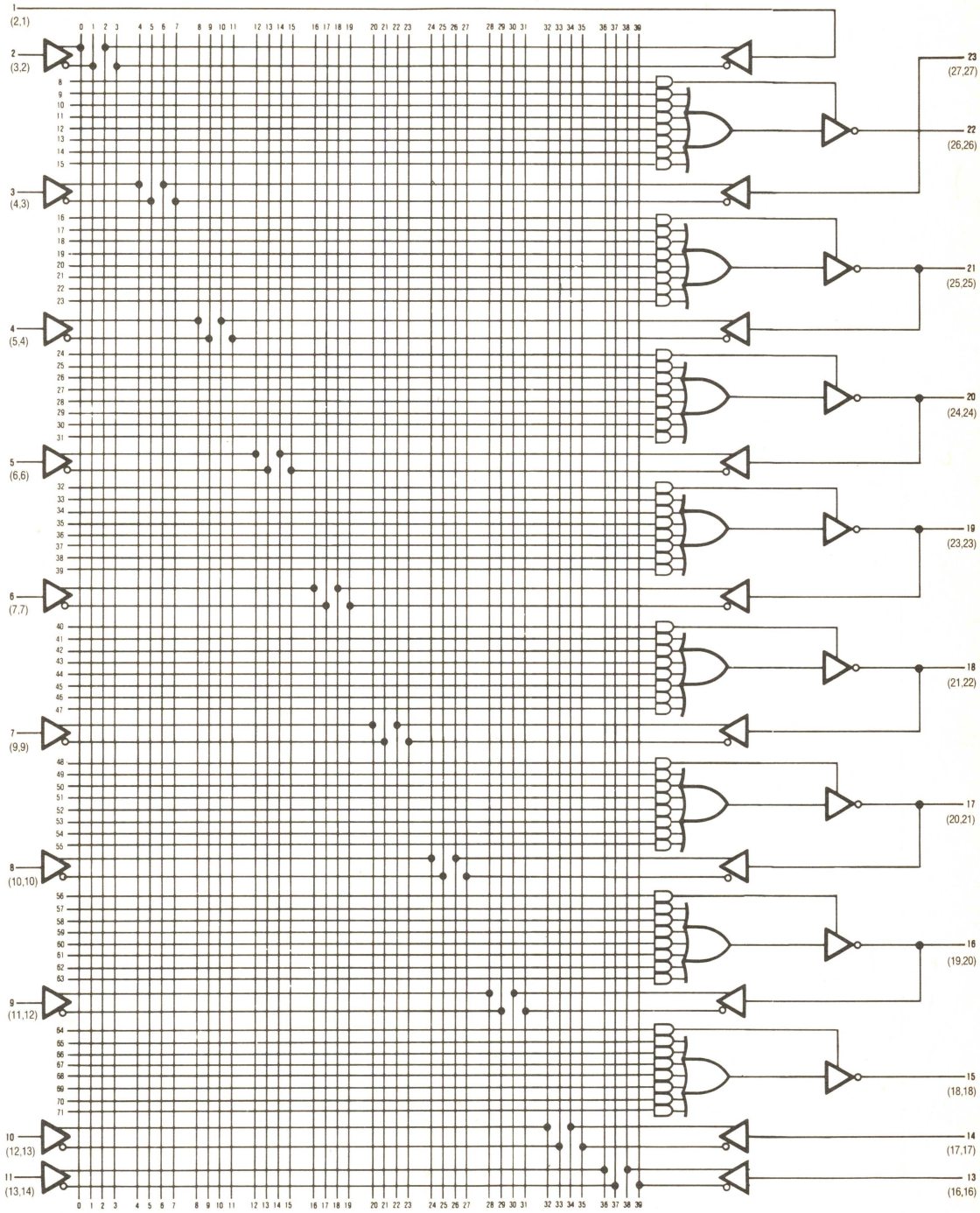
For erasure, the recommended ultraviolet light wavelength is 2537 Angstroms. The minimum dose required is 72,000 mW-sec/cm$^2$ (UV intensity x exposure time). For an ultraviolet lamp with a 20 mW/cm$^2$ power rating, the minimum exposure time would be 72,000/20 seconds = 60 minutes. The device needs to be within one inch of the lamp during erasure.

Permanent damage may result if the device is exposed to high-intensity UV light for an extended period of time. The recommended maximum dosage is 7258 W-sec/cm$^2$.

Wavelengths of light less than 4000 Angstroms can partially erase the device in the windowed package. For this reason, an opaque label should be placed over the window, especially if the device will be exposed to sunlight or fluorescent lighting for extended periods of time.

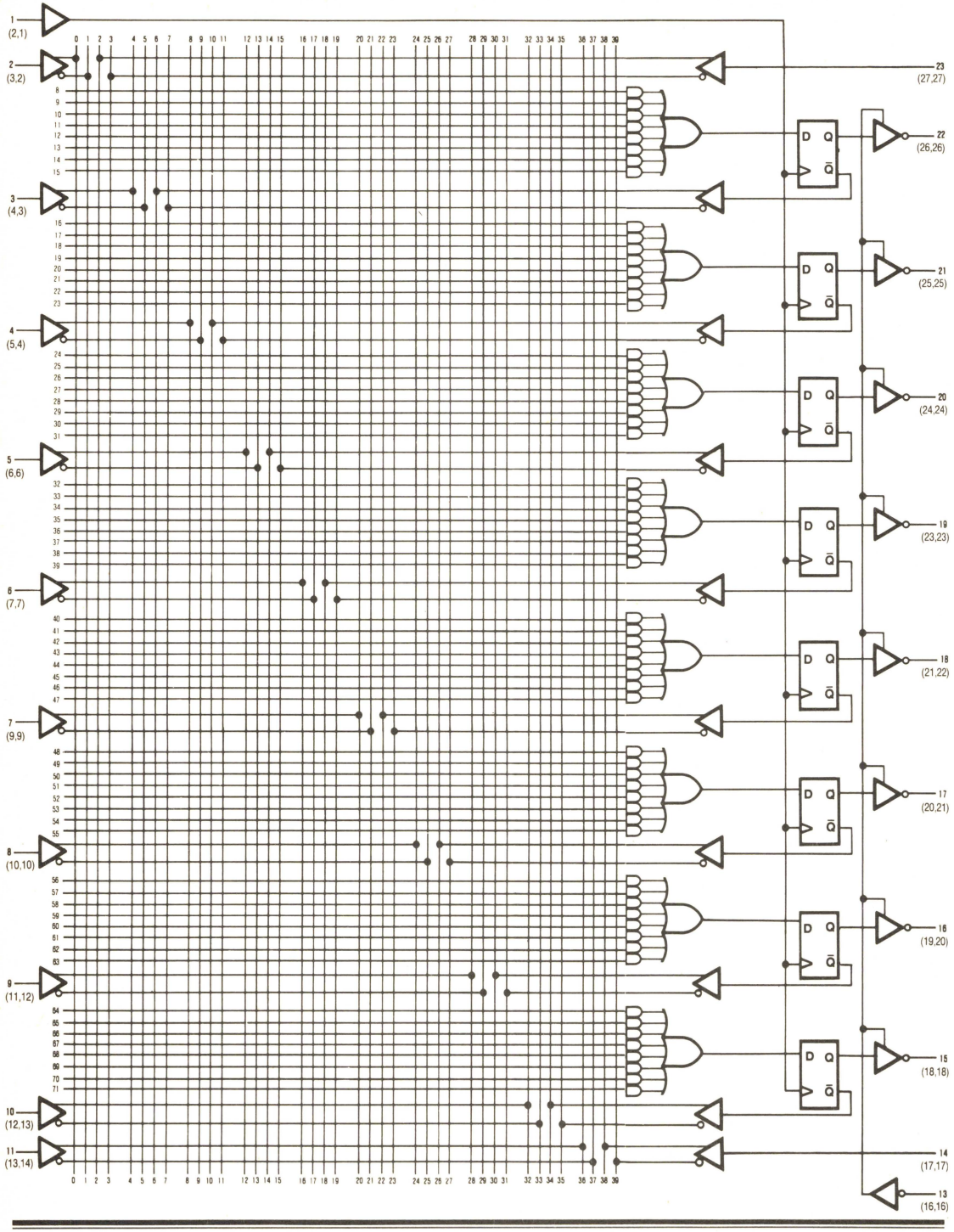
## Logic Diagram
## DIP (FN, NL) Pinouts

20L8



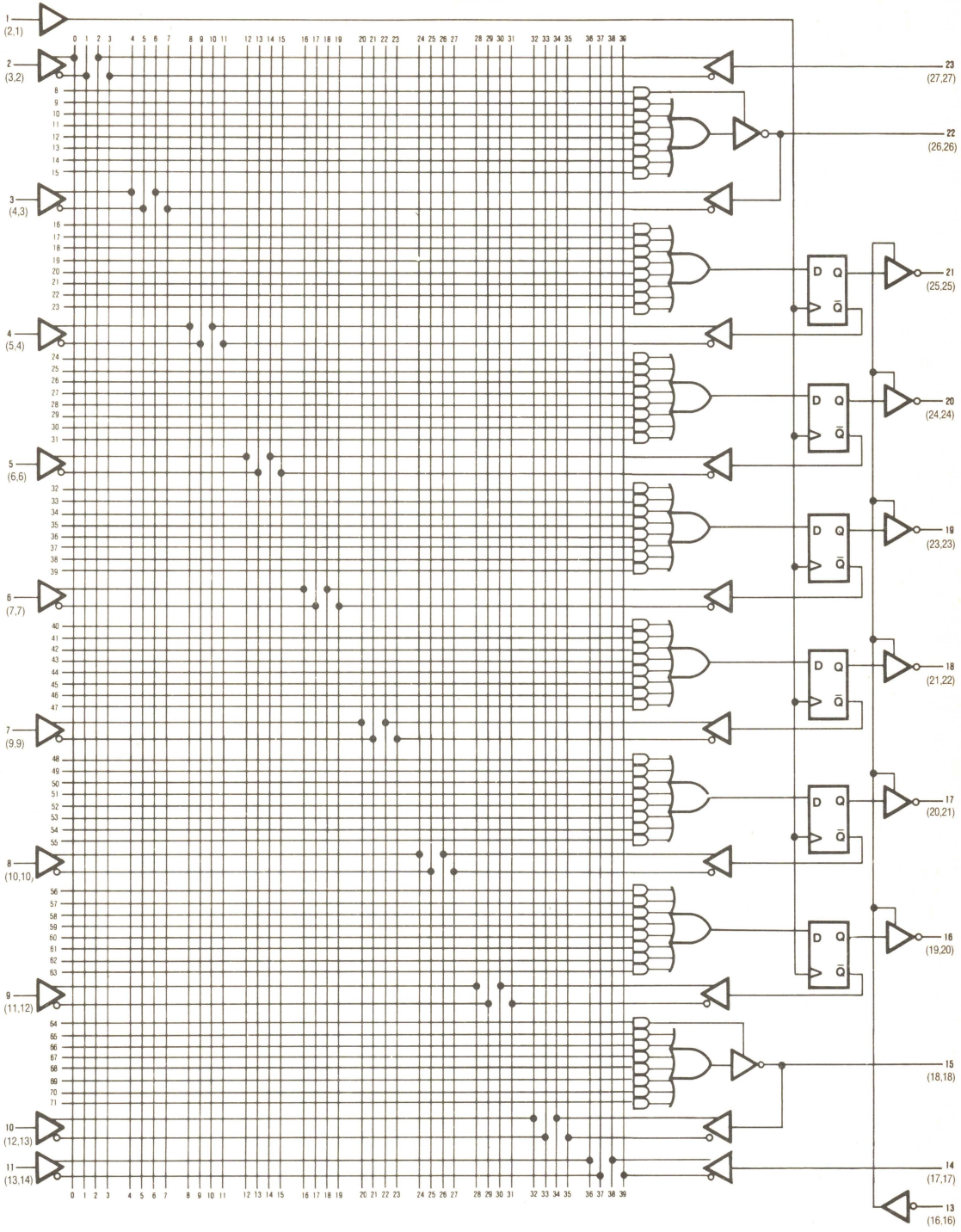

5

## Logic Diagram DIP (FN, NL) Pinouts 20R8



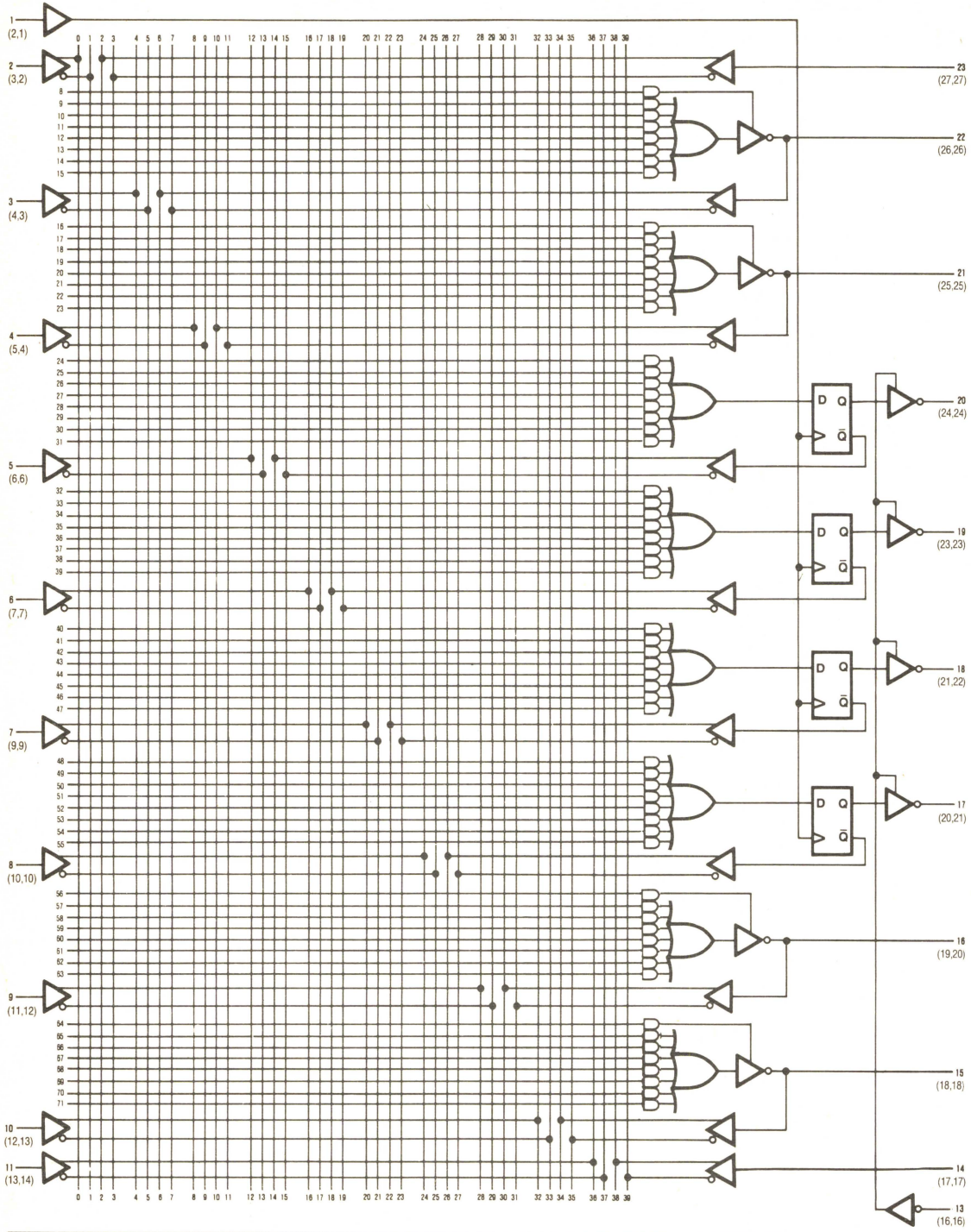
## Logic Diagram DIP (FN, NL) Pinouts

20R6




5

## Logic Diagram DIP (FN, NL) Pinouts 20R4



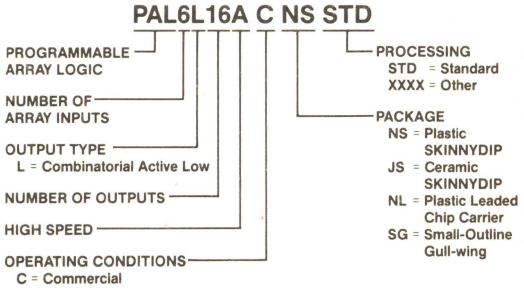
# Decoder Series

# 6L16A
# 8L14A

## Features/Benefits

- 14 to 16 outputs
- Efficient implementation of decoders
- Security fuse

## Ordering Information




## PAL6L16A, 8L14A

| DEVICE | INPUTS | OUTPUTS | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|
| PAL6L16A | 6 | 16 | 25 | 90 |
| PAL8L14A | 8 | 14 | 25 | 90 |

## Description

The Decoder Series provides a wide number of outputs, especially useful in decoding applications. These two parts implement simple combinatorial logic.

## Performance

These devices offer 25-ns speed at only 90 mA supply current.

## Packages

The commercial PAL12L10 Series is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), plastic leaded chip carrier (NL), and small outline (SG) packages.

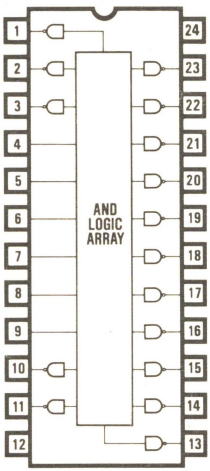
## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

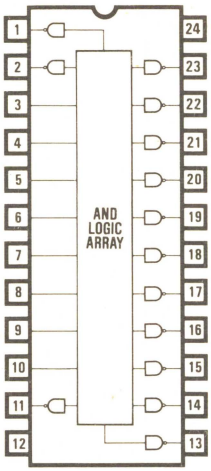
5

## DIP/SO Pinouts

6L16A

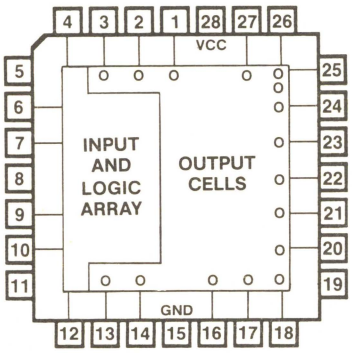



8L14A

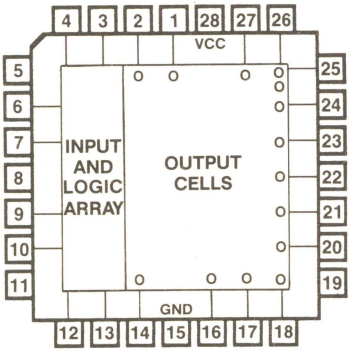



## PLCC Pinouts

6L16A




8L14A



## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | COMMERCIAL[1] MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.75 | 5 | 5.25 | V |
| $T_A$ | Operating free-air temperature | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 60 | 90 | mA |

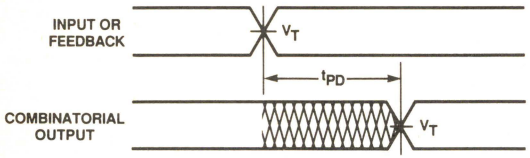
## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{PD}$ | Input to output propagation delay | R1 = 560 Ω, R2 = 1.1 KΩ | | 15 | 25 | ns |

1. The Decoder Series is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

5

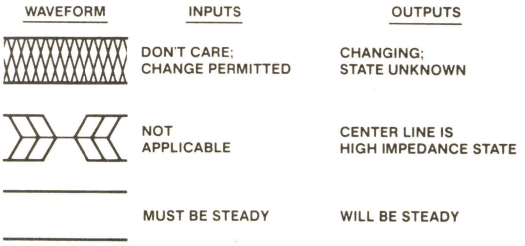
## Switching Waveforms




**Combinatorial Output**

Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.

## Key to Timing Diagrams




## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

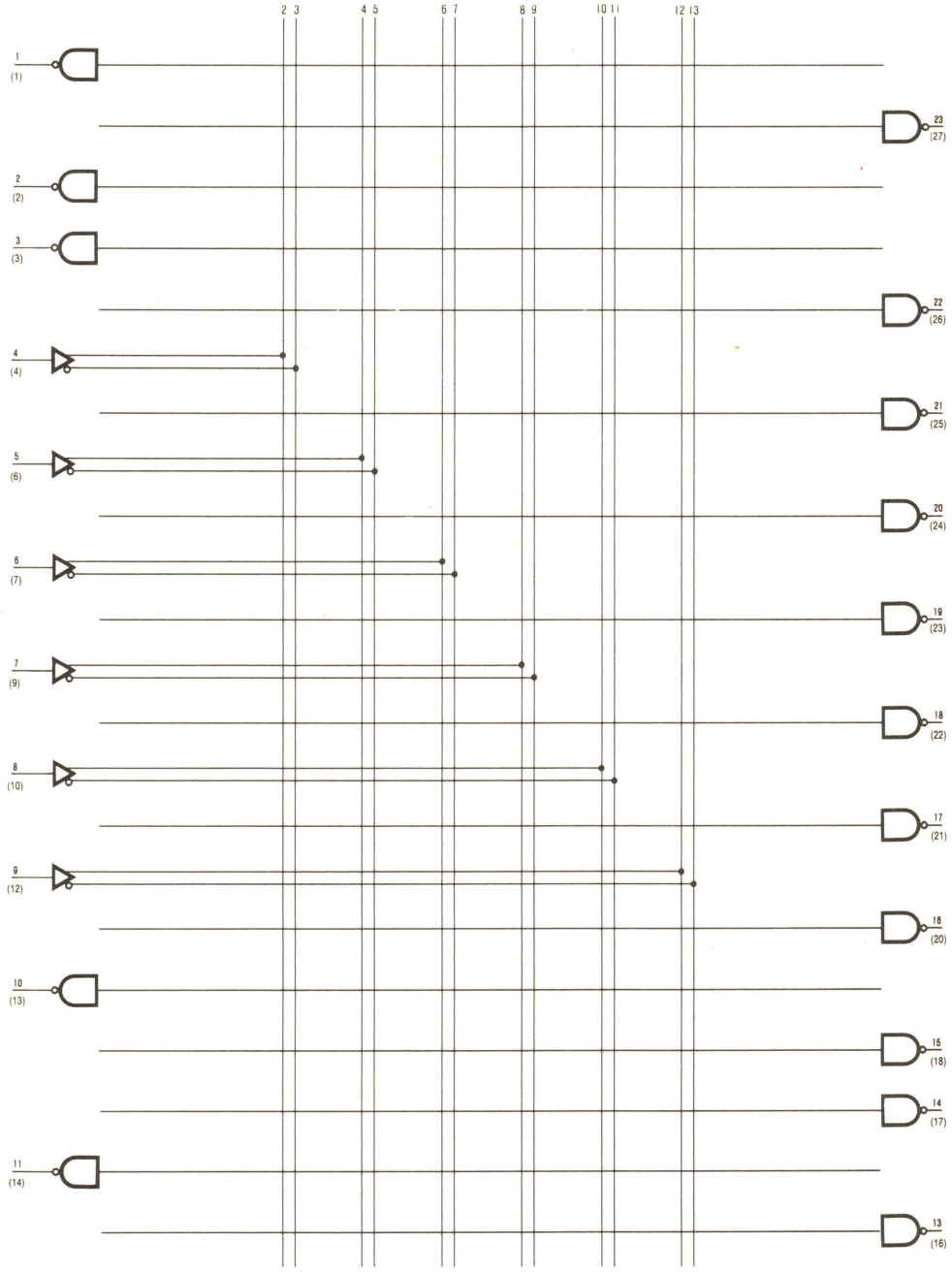
(refer to Programmer Reference Guide, page 3-81)

## Schematic of Inputs and Outputs

(refer to page 5-164)

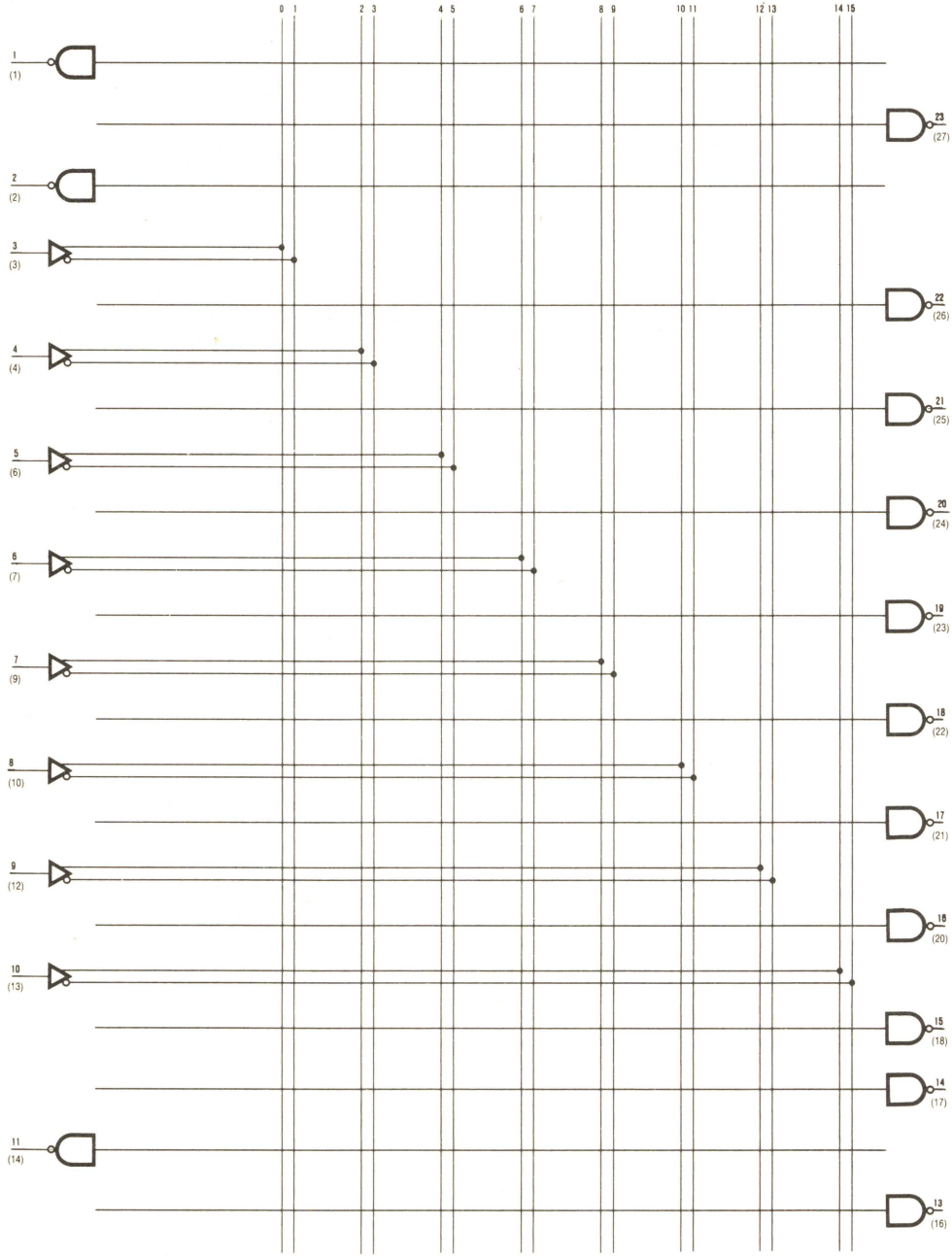
## Logic Diagram

6L16A



## Logic Diagram

8L14A



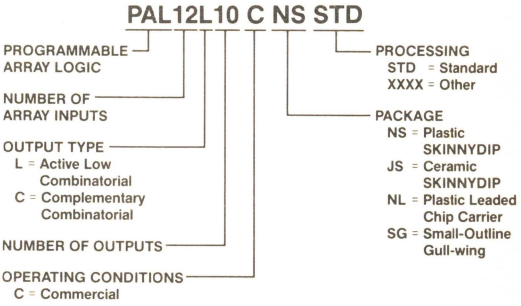
# Combinatorial PAL12L10 Series

# 12L10, 14L8, 16L6 18L4, 20L2, 20C1

## Features/Benefits

- Combinatorial architecture
- Security fuse

## Ordering Information




## PAL12L10 Series

| DEVICE | INPUTS | OUTPUTS | POLARITY | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|---|
| PAL12L10 | 12 | 10 | LOW | 40 | 100 |
| PAL14L8 | 14 | 8 | LOW | 40 | 100 |
| PAL16L6 | 16 | 6 | LOW | 40 | 100 |
| PAL18L4 | 18 | 4 | LOW | 40 | 100 |
| PAL20L2 | 20 | 2 | LOW | 40 | 100 |
| PAL20C1 | 20 | 2 | BOTH | 40 | 100 |

## Description

The PAL12L10 Series is made up of six combinatorial 24-pin PAL devices. They implement simple combinatorial logic, with no feedback.

## Performance

The standard series has a propagation delay (tPD) of 40 nanoseconds (ns). Standard supply current is 100 milliamps (mA).

## Packages

The commercial PAL12L10 Series is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), plastic leaded chip carrier (NL), and small outline (SG) packages.

## Package Drawings

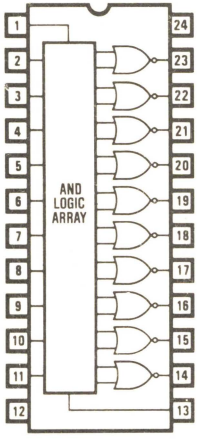
(refer to PAL Device Package Outlines, page 3-179)



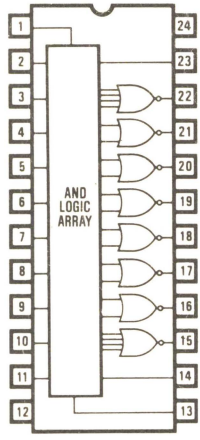
10326A
JANUARY 1988

## DIP/SO Pinouts

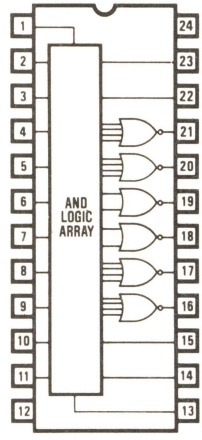
12L10



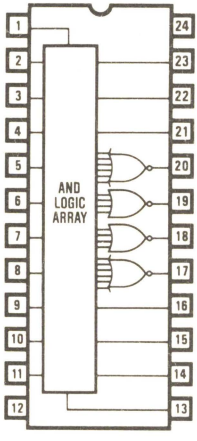

14L8



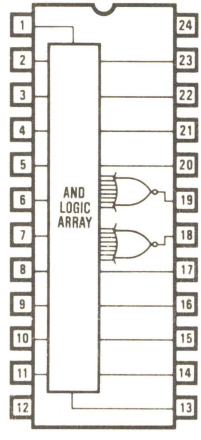

16L6



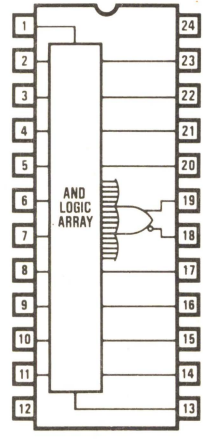

18L4




20L2

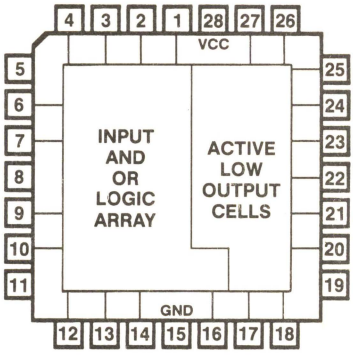



20C1

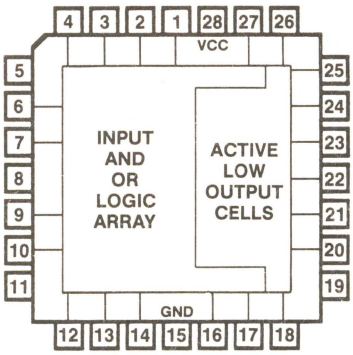


## PLCC Pinouts

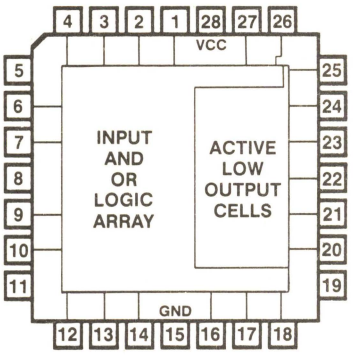
12L10



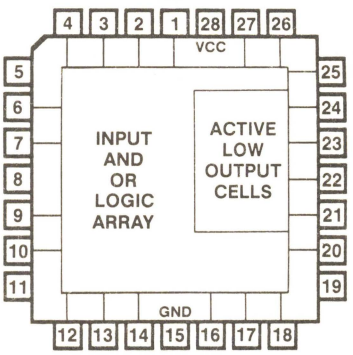

14L8



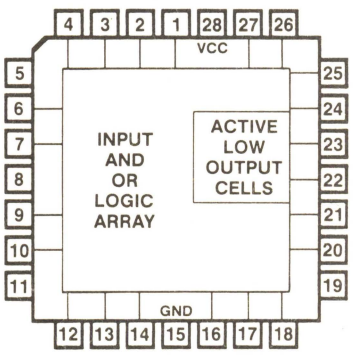

16L6



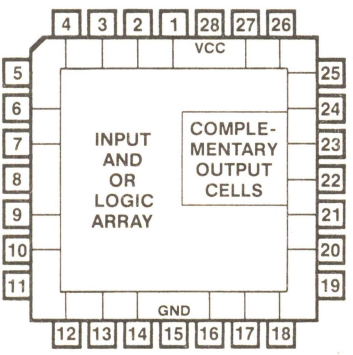

18L4




20L2




20C1




5

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65° C to +150° C | |

## Operating Conditions

| SYMBOL | PARAMETER | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.75 | 5 | 5.25 | V |
| $T_A$ | Operating free-air temperature | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

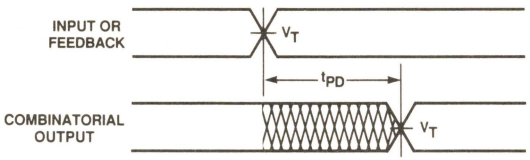
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 2.8 | | V |
| $I_{OS}$[2] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 60 | 100 | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | R1 = 560 Ω, R2 = 1.1 KΩ | | 25 | 40 | ns |

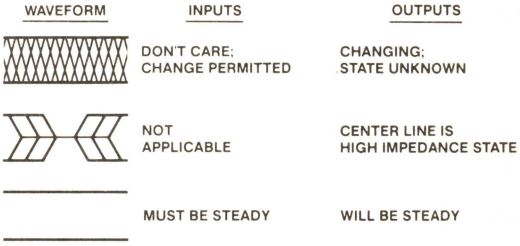
1. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

## Switching Waveforms




Combinatorial Output

## Key to Timing Diagrams




Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.

## Switching Test Load

(refer to page 5-164)

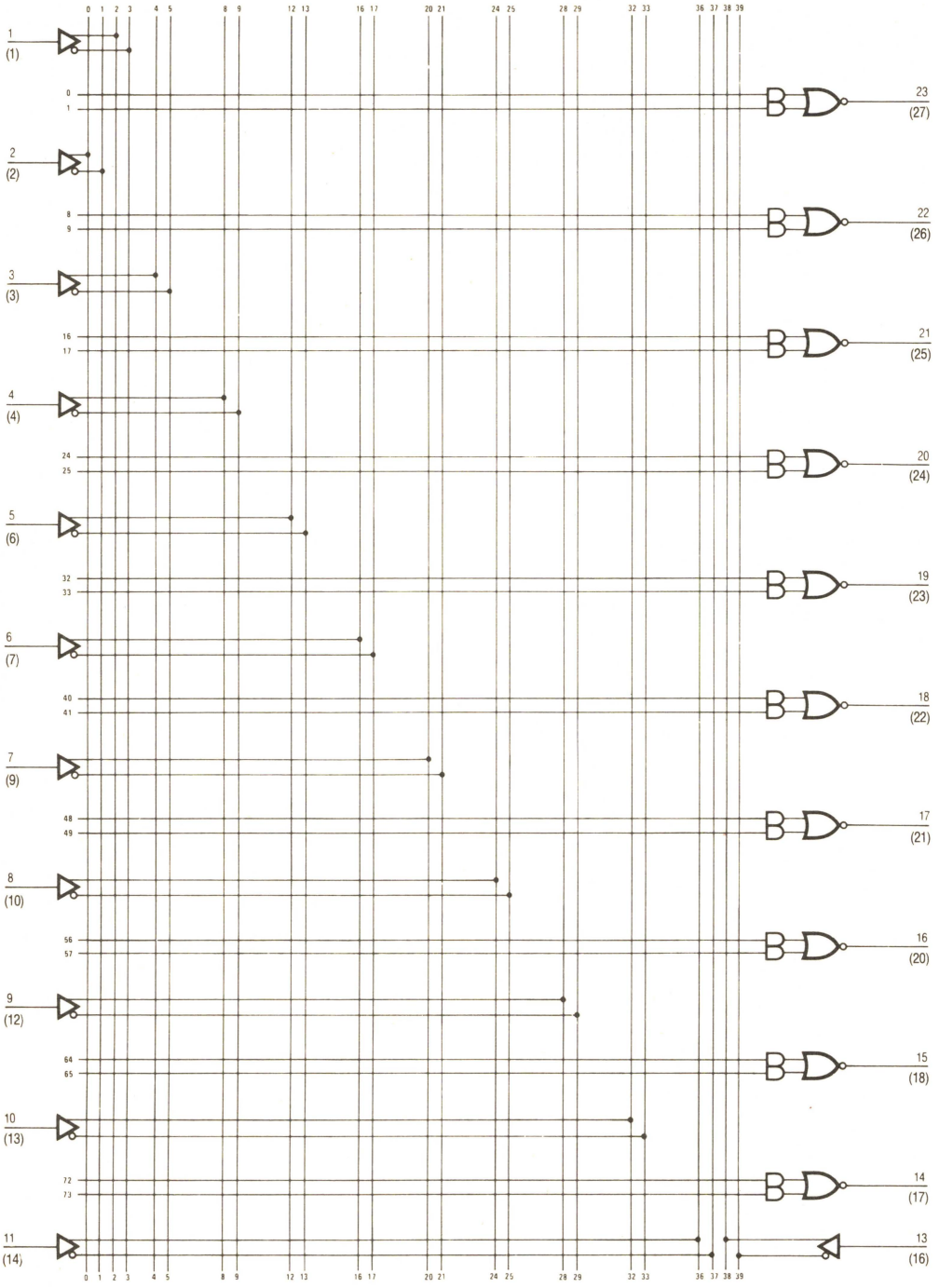
## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

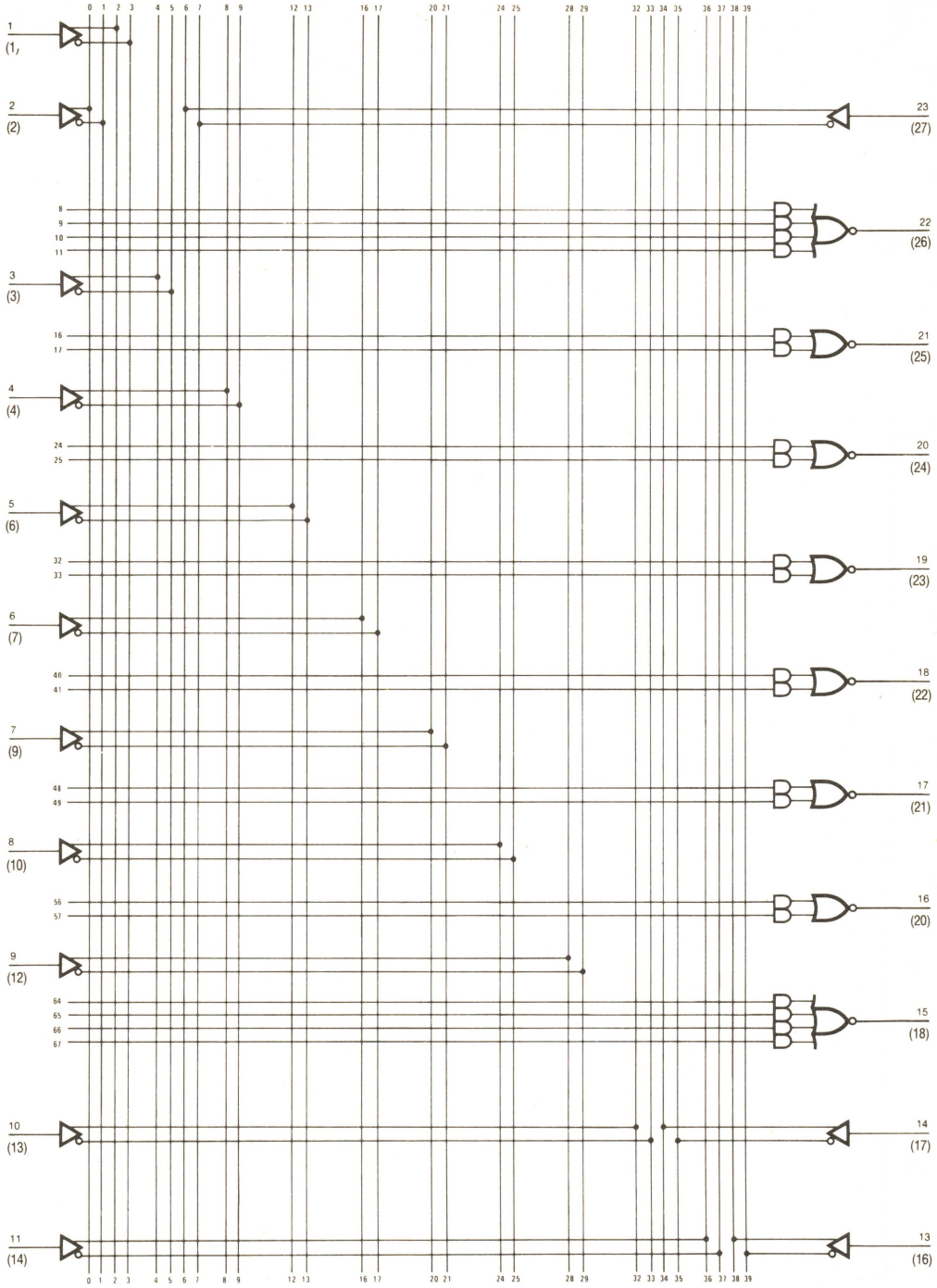
## Schematic of Inputs and Outputs

(refer to page 5-164)

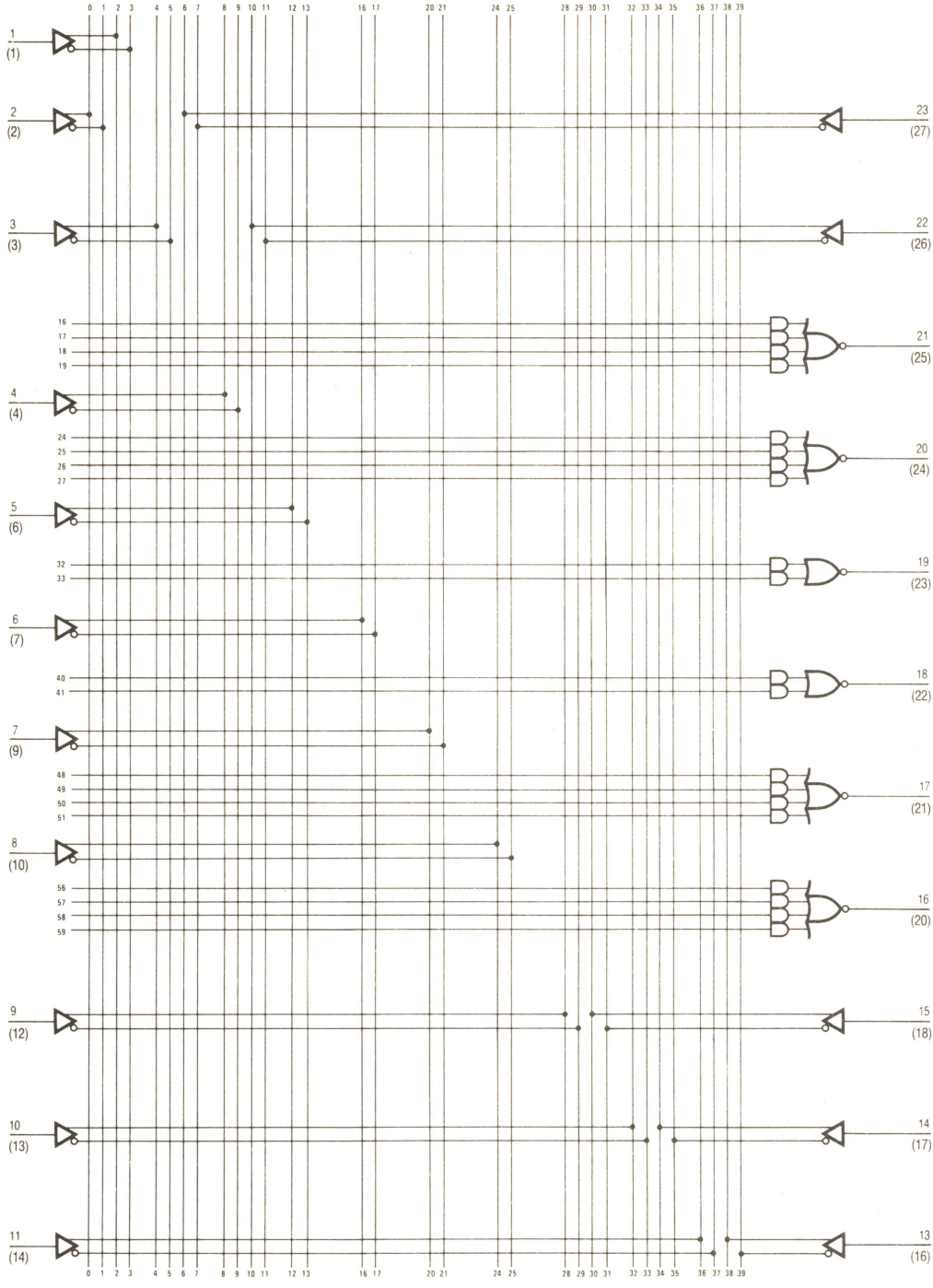
## Logic Diagram DIP (PLCC) Pinouts 12L10



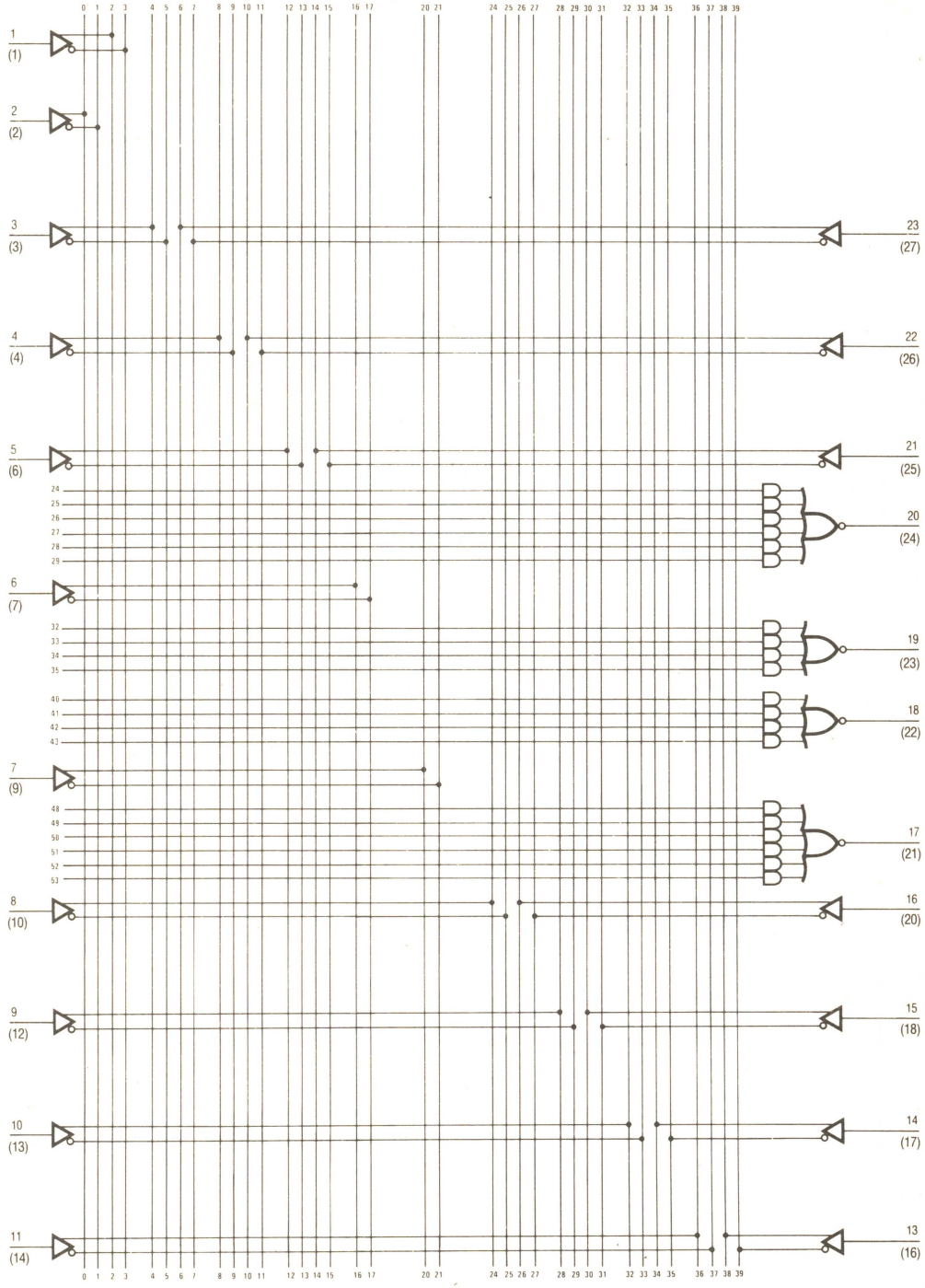
## Logic Diagram DIP (PLCC) Pinouts 14L8



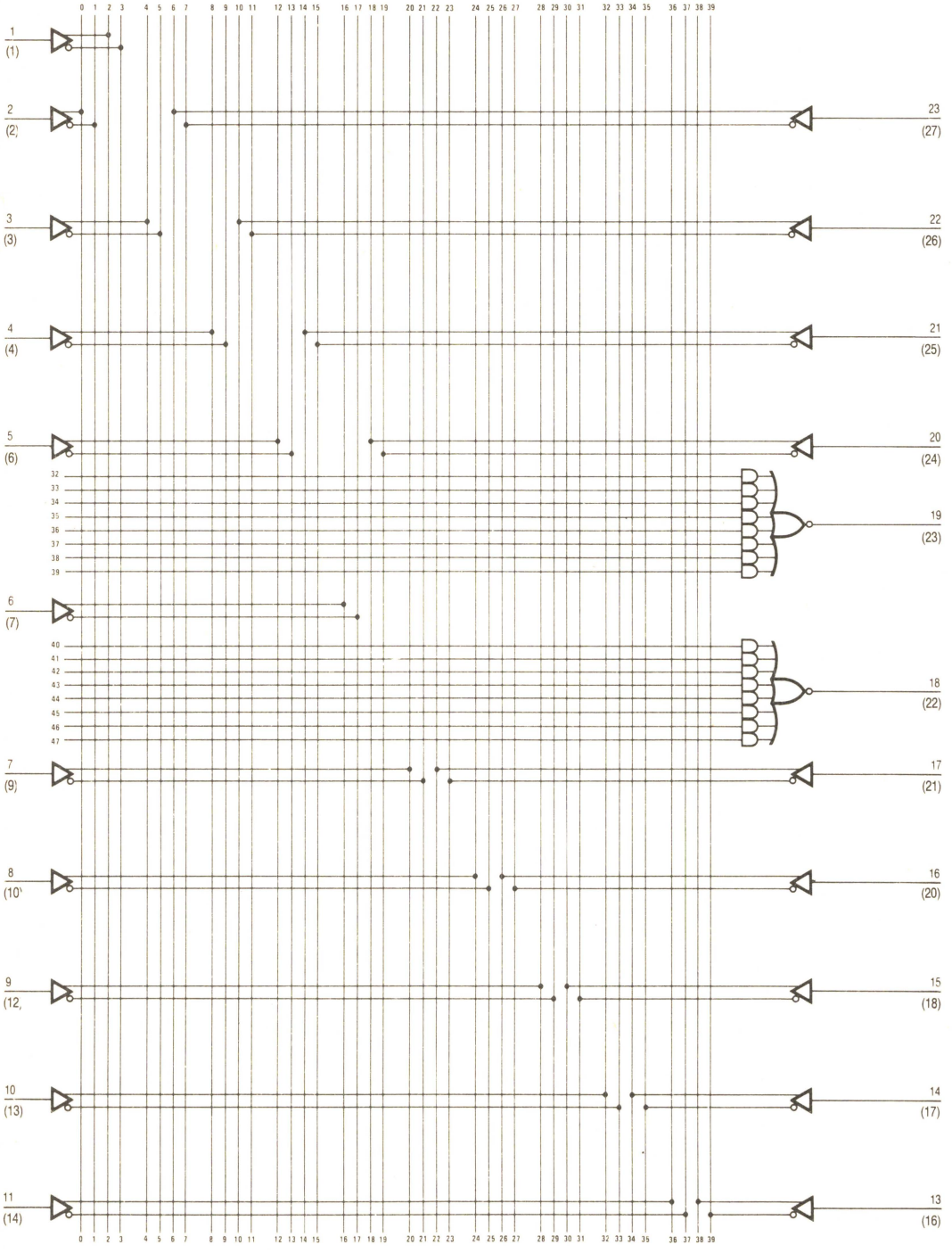
## Logic Diagram DIP (PLCC) Pinouts 16L6



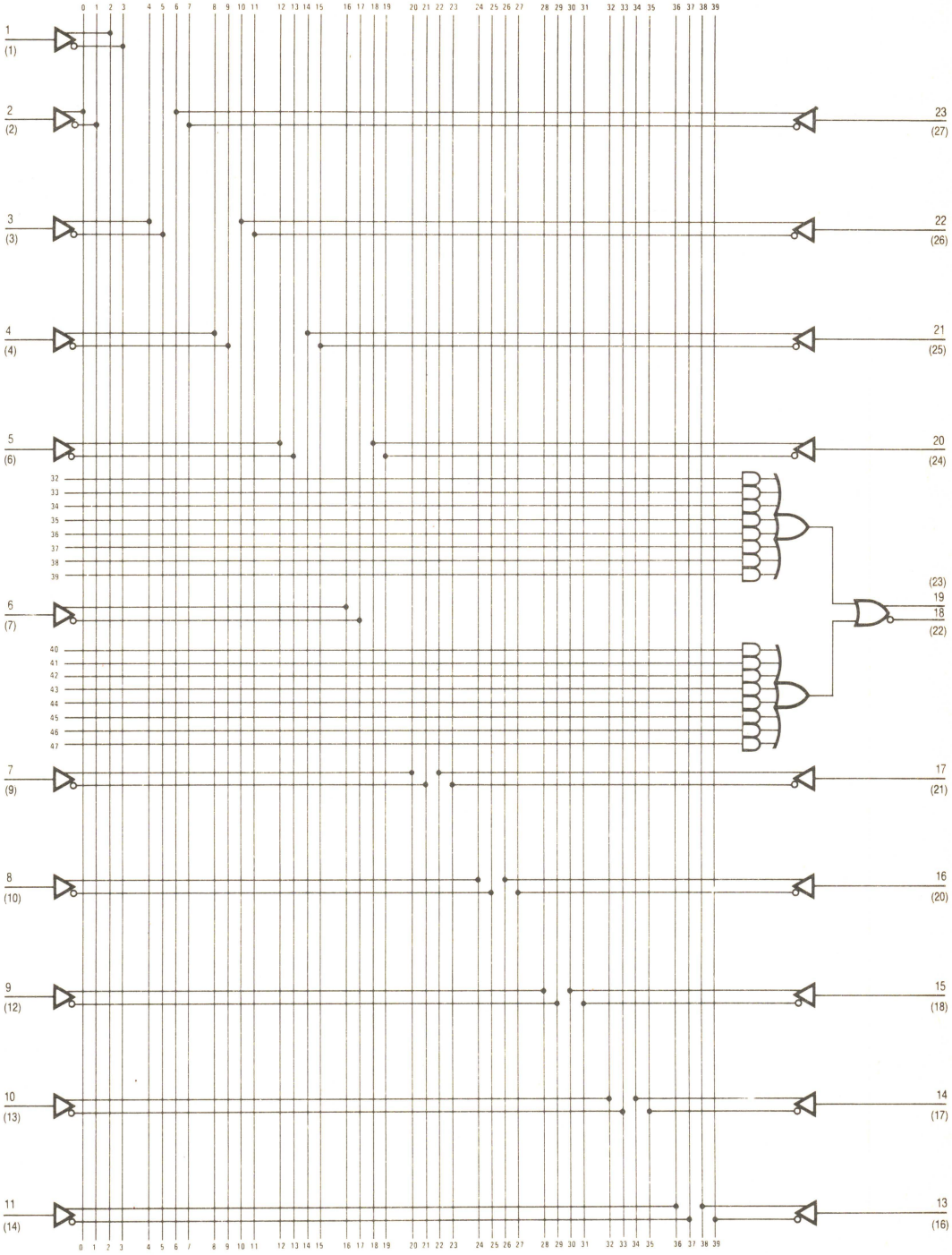
## Logic Diagram DIP (PLCC) Pinouts 18L4



## Logic Diagram DIP (PLCC) Pinouts 20L2



## Logic Diagram DIP (PLCC) Pinouts 20C1

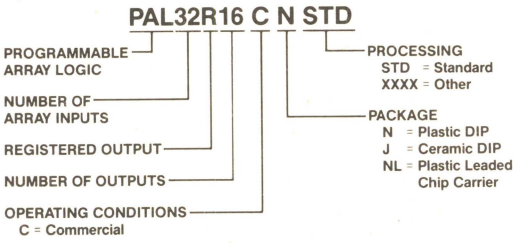


# MegaPAL Device
# PAL32R16

## Features/Benefits

- High-density 40-pin architecture
- Product term steering allows up to 16 product terms per output
- Programmable polarity
- Register bypass in banks of 8
- TTL level register preload
- Power-up reset
- Security fuse

## Ordering Information




## MegaPAL Device

| | ARRAY INPUTS | REGISTERED INPUTS | $t_{PD}$ (ns) | $I_{CC}$ (mA) |
|---|---|---|---|---|
| PAL32R16 | 32 | 16 | 40 | 280 |

## Description

The MegaPAL Device offers very high density programmable logic.

The PAL device transfer function is the familiar Boolean sum of products. The PAL device consists of a programmable AND array driving a fixed OR array. Product terms with all bits programmed (disconnected) assume the logical high state, and product terms with both true and complement of any signal connected assume the logical low state.

## Variable Input/Output Pin Ratio

Each combinatorial output is an I/O pin. Buffers for device inputs have complementary outputs to provide user-programmable input signal polarity. Unused input pins should be tied directly to VCC or GND.

## Programmable Three-State Outputs

Each output has a three-state output buffer with programmable three-state control. The output provides a bidirectional I/O pin in the combinatorial configuration, and may be configured as a dedicated input if the buffer is always disabled.

## Registers with Feedback

Registered outputs are provided for data storage and synchronization. Registers are composed of D-type flip-flops which are loaded on the low-to-high transition of the clock input.

## Package Drawings

(refer to PAL Device Package Outlines, page 3-179)

## Programmable Polarity

Each flip-flop has individually programmable polarity. The unprogrammed state is active low.

## Product Term Steering

Product term steering allows each pair of outputs to share its product terms with one output or the other (not both). Each pair has a total of sixteen product terms; thus, one output can use zero to sixteen terms while the other has sixteen to zero. Product terms can only be shared mutually exclusively. If both output need the same term, it must be created twice, once for each output.

## Register Bypass

Registers can be bypassed in banks of eight, creating a set of combinatorial outputs.

## Preload and Power-Up Reset

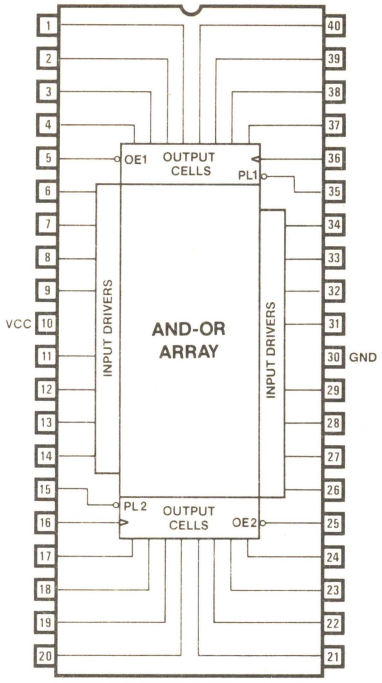
The device also offers register preload for device testability. The registers can be preloaded from the outputs by using TTL level signals in order to simplify functional testing. This series also offers Power-Up Reset, whereby the registers power up to a logic LOW, setting the active-low outputs to a logic HIGH.

## Packages

The commerical PAL32R16 is available in the plastic and ceramic DIP (N,J) and 44-pin plastic leaded chip carrier (NL) packages.

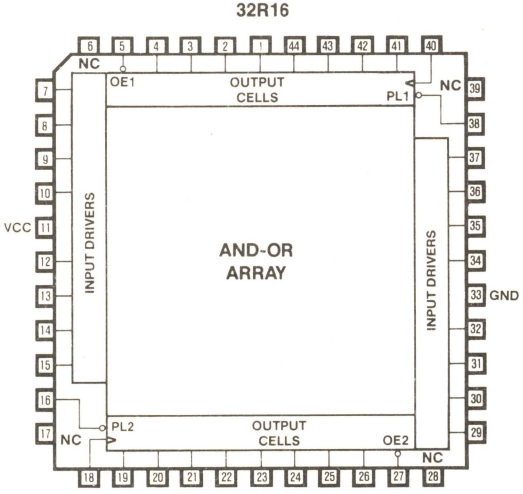
10329A
JANUARY 1988

32R16




DIP

32R16




Plastic Chip Carrier

5

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 22.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | Low | 20 | | | ns |
| | | High | 20 | | | |
| $t_{wp}$ | Preload pulse width | | 35 | | | ns |
| $t_{su}$ | Set up time for input to clock | Polarity fuse intact | 40 | | | ns |
| | | Polarity fuse programmed | 40 | | | |
| $t_{sup}$ | Preload set up time | | 25 | | | ns |
| $t_h$ | Hold time | | 0 | -10 | | ns |
| $t_{hp}$ | Preload hold time | | 5 | | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

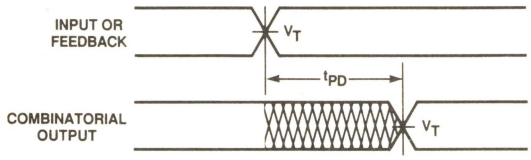
| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$[2] | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$[2] | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 100 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = 3.2 mA | 2.4 | 2.8 | | V |
| $I_{OZL}$[2] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$[2] | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = MAX | $V_O$ = 0 V | -30 | -70 | -130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 200 | 280 | mA |

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. I/O pin leakage is the worst case of $I_{IL}$ and $I_{OZL}$ (or $I_{IH}$ and $I_{OZH}$).
3. No more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.

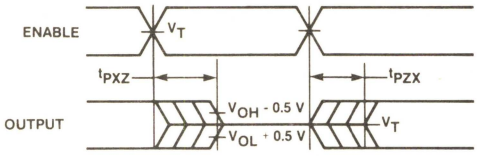
## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input to output | Polarity fuse intact | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | | 40 | ns |
| | | Polarity fuse programmed | | | | 45 | |
| $t_{CLK}$ | Clock to output or feedback | | | | | 25 | ns |
| $t_{PZX}$ | Output enable | | | | | 20 | ns |
| $t_{PXZ}$ | Output disable | | | | | 20 | ns |
| $f_{MAX}$ | Maximum frequency | External | | 16 | | | MHz |
| | | No feedback | | 25 | | | |

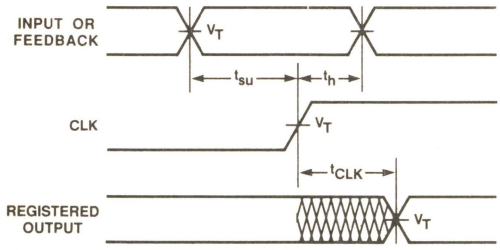
## Switching Waveforms



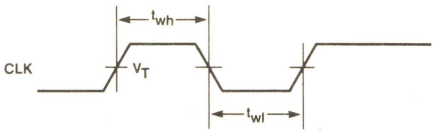

Combinatorial Output




Pin to Output Disable/Enable




Registered Output

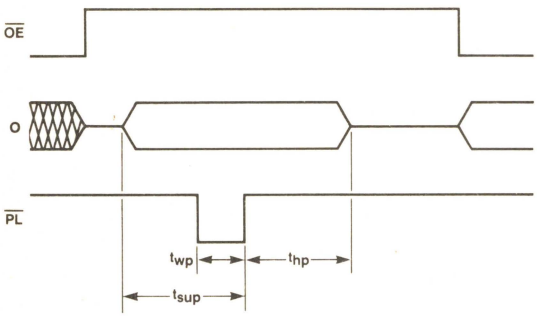



Clock Width

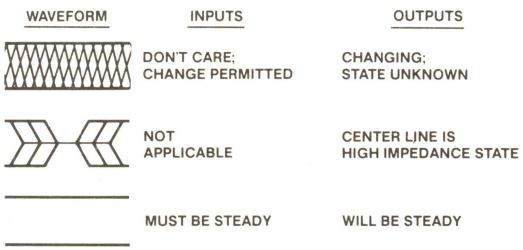
Notes:
1. VT = 1.5 V.
2. Input pulse amplitude 0 V to 3.0 V.
3. Input rise and fall times 2-5 ns typical.

## Register Preload

Register preload allows any arbitrary state to be loaded into the PAL device output registers. This allows complete logic verification, including states that are impossible or impractical to reach. To use the preload feature, first disable the outputs by bringing $\overline{OE}$ high, and present the data at the output pins. A low level on the preload pin ($\overline{PL}$) will then load the data into the registers.




## Key to Timing Diagrams




## Switching Test Load

(refer to page 5-164)

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Power-Up Reset Waveform

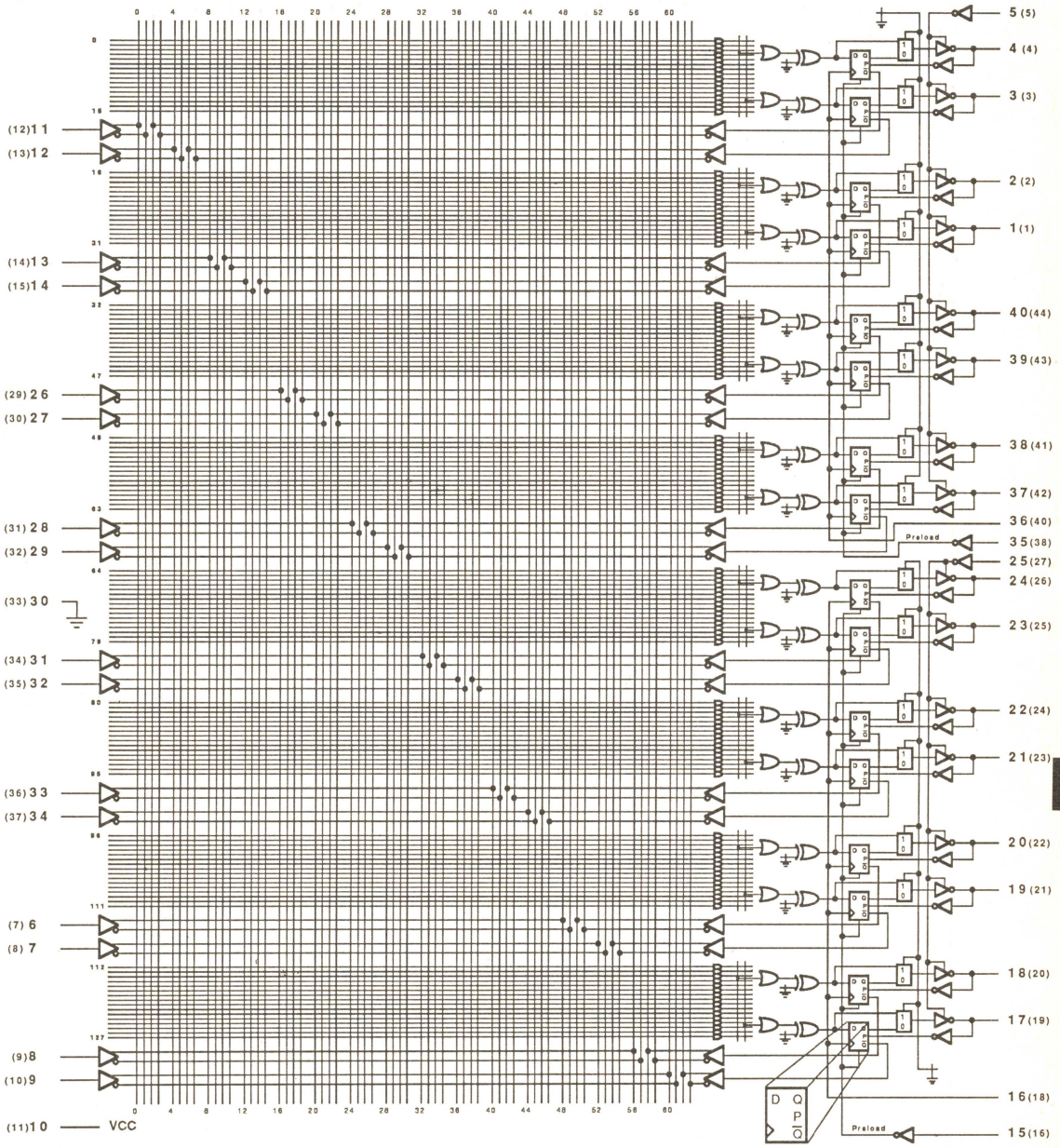
(refer to page 5-164)

## Schematic of Inputs and Outputs

(refer to page 5-164)

## 32R16

Logic Diagram

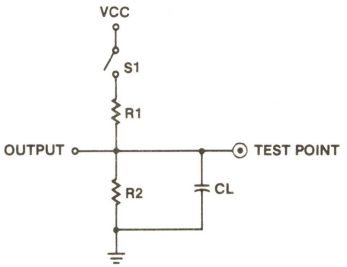



DIP(PLCC) Pinouts

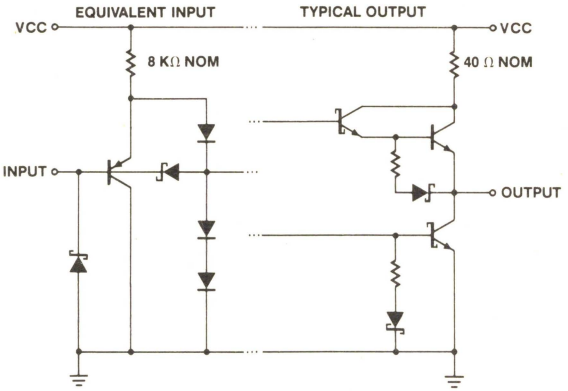
5

# General Information for TTL/CMOS PAL Devices

## Switching Test Load

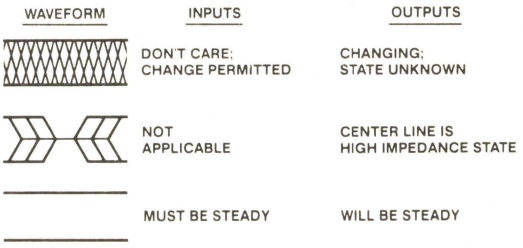



## Schematic of Inputs and Outputs (TTL)




| SPECIFICATION | SWITCH S1 | $C_L$ | MEASURED OUTPUT VALUE |
|---|---|---|---|
| $t_{PD}$, $t_{CLK}$, $t_{CF}$ | Closed | 50 pF | 1.5 V |
| $t_{PZX}$, $t_{EA}$ | Z->H: open<br>Z->L: closed | 50 pF | 1.5 V |
| $t_{PXZ}$, $t_{ER}$ | H->Z: open<br>L->Z: closed | 5 pF | H->Z: $V_{OH}$-0.5 V<br>L->Z: $V_{OL}$+0.5 V |

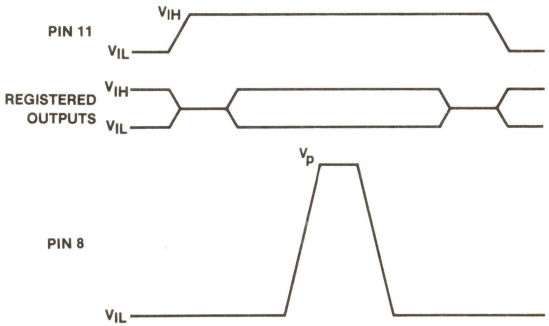
## Key to Timing Diagrams



## Output Register PRELOAD (16RP8A Series)

The PRELOAD function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct setting of output states. The procedure is as follows:

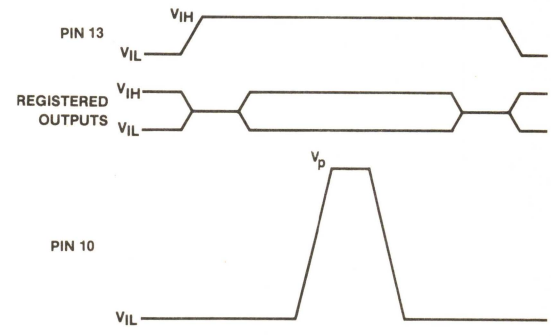
1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 11 to $V_{IH}$.
3. Apply the desired value ($V_{IL}/V_{IH}$) to all registered output pins. Leave combinatorial outputs floating.
4. Pulse pin 8 to $V_P$ (20 V), then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all registered output pins.
6. Lower pin 11 to $V_{IL}$ to enable the output registers.
7. Verify for $V_{OL}/V_{OH}$ at all registered output pins. Note that because of the output inverter, a preloaded HIGH will set the register HIGH, providing a LOW at the output.




## Output Register PRELOAD (20R8B-2 Series, 20X10A Series, 20RS10 Series)

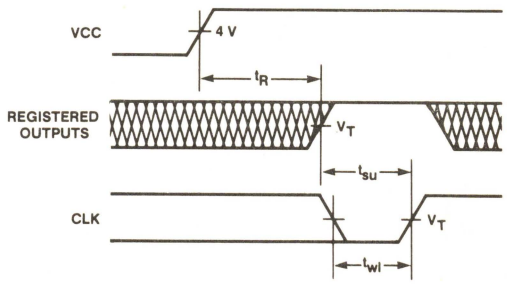
The PRELOAD function allows the register to be loaded from the output pins. This feature aids functional testing of sequential designs by allowing direct setting of output states. The procedure is as follows:

1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 13 to $V_{IH}$.
3. Apply the desired value ($V_{IL}/V_{IH}$) to all registered output pins. Leave combinatorial outputs floating.
4. Pulse pin 10 to $V_P$ (20 V), then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all registered output pins.
6. Lower pin 13 to $V_{IL}$ to enable the output registers.
7. Verify for $V_{OL}/V_{OH}$ at all registered output pins. Note that because of the output inverter, a preloaded HIGH will set the register HIGH, providing a LOW at the output.




## Power-Up Reset

The power-up reset function forces the register to the logic LOW state on the application of voltage to the VCC pin. Active-LOW outputs will be HIGH because of the output inverter. This feature allows the device to always power-up to a known state, for predictable system initialization. The power-up reset time, tR, is 1 $\mu$s maximum. Following reset, minimum setup times and clock widths must be met.




5

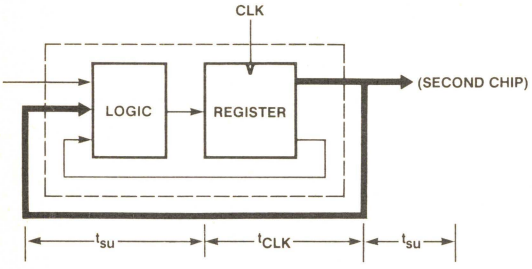
## $f_{MAX}$ Parameters

The parameter fMAX is the maximum clock rate at which the device is guaranteed to operate. Because flexibility inherent in programmable logic devices offers a choice of clocked flip-flop designs, fMAX is specified for three types of synchronous designs.

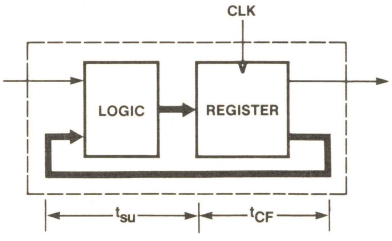
The first type of design is a state machine with feedback signals sent off-chip. This external feedback could go back to the device inputs, or to a second device in a multi-chip state machine. The slowest path defining the period is the sum of the clock-to-output time and the input setup time for the external signals (tsu + tCLK). The reciprocal, fMAX, is the maximum frequency with external feedback or in conjunction with an equivalent speed device. This fMAX is designated "fMAX external."

The second type of design is a single-chip state machine with internal feedback only. In this case, flip-flop inputs are defined by the device inputs and flip-flop outputs. Under these conditions, the period is limited by the internal delay from the flip-flop outputs through the internal feedback and logic to the flip-flop inputs (tsu + tCF). This fMAX is designated "fMAX internal."

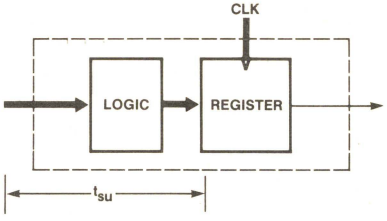
The third type of design is a simple data path application. In this case, input data is presented to the flip-flop and clocked through; no feedback is employed. Under these conditions, the period is limited by the sum of the data setup time and the data hold time (tsu + th). However, a lower limit for the period of each fMAX type is the minimum clock period (twh + twl). Usually, this minimum clock period determines the period for the third fMAX, designated "fMAX no feedback."



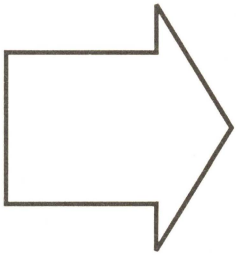

$f_{MAX}$ EXTERNAL; $1/(t_{su}+t_{CLK})$




$f_{MAX}$ INTERNAL; $1/(t_{su}+t_{CF})$




$f_{MAX}$ NO FEEDBACK; $1/(t_{su}+t_h)$ or $1/(t_{wh}+t_{wl})$



## Data Sheets


TTL/CMOS PAL Devices

TTL/CMOS AmPAL Devices

PROSE/PLS Sequencers

FPC/PEG Sequencers

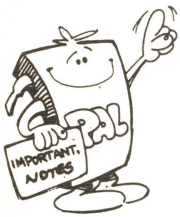
ECL PAL Devices

HAL/ZHAL Devices

Military PAL Devices

Logic Cell Array

Electrical Definitions


IMPORTANT
NOTES

# AmPAL23S8

20-Pin IMOX™ PAL Device-Based Sequencer

## Distinctive Characteristics

- 14 Registers
  - 4 Output Logic Macrocells (OLMs)
  - 4 Output Registers
  - 6 Buried State Registers (BSRs)
- 23 possible array inputs and 8 outputs in a 20-pin package
- 33-MHz external/40-MHz internal cycle time
- Variable product term (PT) distribution for increased design flexibility
- Asynchronous and synchronous outputs supported for both Mealy- and Moore-type state-machine implementations
- Individually user-programmable Output Enable (OE) PTs with polarity control
- PTs for observing the BSRs on 6 of the output pins
- Separate PTs for common Synchronous PRESET and common Asynchronous RESET of all registers
- PRELOAD available on all registers for added test capability
- 99.9% post programming functional yield (PPFY) is due to the testability of this and all other AMD PAL devices.
- Platinum-Silicide fuse technology produces the most reliable bipolar programmable devices available today

## General Description

The AmPAL23S8 is the first programmable array logic (PAL)-based sequencer device. It utilizes the familiar sum-of-products (AND-OR) logic structure, allowing users to customize logic functions by programming the device for specific applications. The AmPAL23S8 combines the ease of use of the familiar 20-pin PAL devices with the advanced "macrocell" concept introduced in the AmPAL22V10, as well as six Buried State Registers (BSRs).

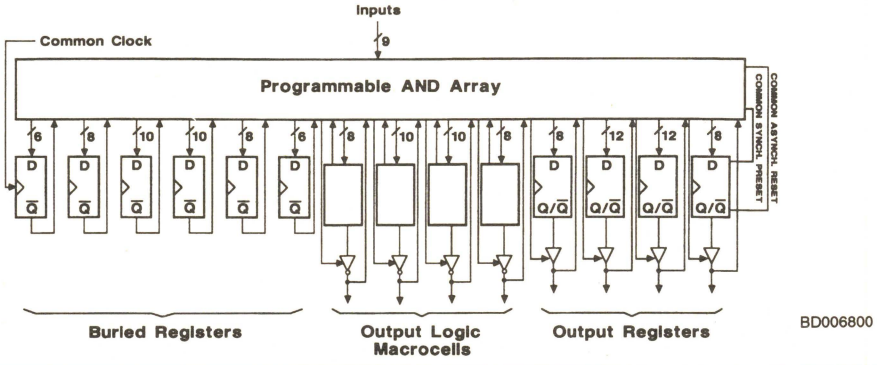
The AmPAL23S8 provides up to twenty-three array inputs and eight outputs. Four of the outputs are Output Logic Macrocells (OLMs) capable of being individually programmed as "combinatorial" or "registered," with active-HIGH or active-LOW polarity on each output. The other four are "registered" outputs, also capable of being programmed for active- HIGH or LOW polarity. All the flexibility on the outputs result in the simplification of logic design. The need to perform "DeMorgan's Law" on equations to have them fit into a PAL device is now a thing of the past. Each of the eight output registers can also be used dynamically as an input or output for greater design flexibility.

The AmPAL23S8 also offers designers increased flexibility and control over Output Enable (OE) functions. Each output is logically controlled by an OE product term (PT), with programmable OE polarity control. This allows the designer to use more complex control than previously available.

The six BSRs provide designers with enhanced logic power for sequencer applications. These registers are not only available to the system designer for use in sequencer applications (without the expense of a valuable I/O pin), but they may also be observed on the output pins during test. The observability of these registers on a programmable-logic sequencer adds to the list of features which make this device unique, simple to design with, and simple to debug.

System operation has been enhanced by the addition of Synchronous PRESET and Asynchronous RESET PTs. The AmPAL23S8 also incorporates the unique capability of PRELOADing the eight output registers and the BSRs to any desired state during testing. This is essential to permit full logical verification during test.

## Block Diagram



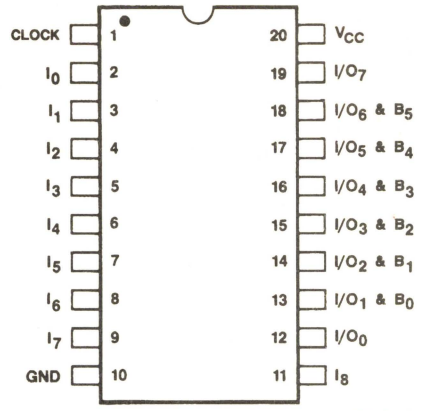

BD006800

5

08207D/0
JANUARY 1988

## Connection Diagram

Top View




CD009870

Note: Pin 1 is marked for orientation.

### Pin Description

$V_{CC}$ = Supply Voltage

GND = Ground

CLOCK = Clock Pin

$I_0 - I_8$ = Dedicated Input Pins (9)

$I/O_0 - I/O_7$ = Bidirectional I/O Pins (8)

$B_0 - B_5$ = Observability Pins for BSRs (6)

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**

AMPAL23S8 –20 P C —

**E. OPTIONAL PROCESSING**
Blank = Standard processing
B = Burn-in

**D. TEMPERATURE RANGE**
C = Commercial (0 to +70°C)
E = Extended Commercial (–55 to +125°C)

**C. PACKAGE TYPE**
P = 20-Pin Plastic DIP (PD 020)
D = 20-Pin Ceramic DIP (CD 020)

**B. SPEED OPTION**
–20 = 20 ns
–25 = 25 ns
–27 = 27 ns*
–30 = 30 ns*

**A. DEVICE NUMBER/DESCRIPTION**
AmPAL23S8
20-Pin IMOX PAL-Based Sequencer

*Extended Commercial range only.

| Valid Combinations | |
|---|---|
| AMPAL23S8-20 | PC, DC, DCB |
| AMPAL23S8-25 | PC, DC, DCB, DE, DEB |
| AMPAL23S8-27 | DE, DEB |
| AMPAL23S8-30 | |

**Valid Combinations**

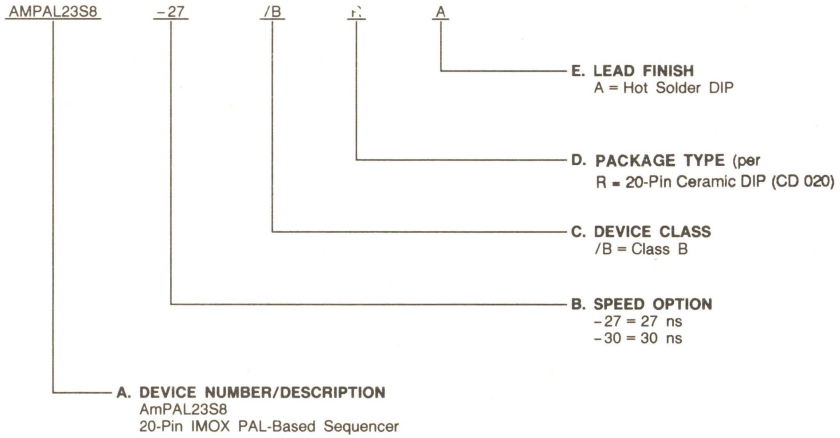
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information (Cont'd.)

### APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL23S8-27 | /BRA |
| AMPAL23S8-30 | |

#### Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

#### Group A Tests

Group A Tests consist of Subgroups: 1, 2, 3, 7, 8, 9, 10, 11

5

## Functional Description

The AmPAL23S8 is an advanced bipolar programmable array logic (PAL)-based sequencer. It contains a programmable array organized in the familiar sum-of-products structure. The structure of this device makes it particularly ideal for state machine applications. Any design which employs the use of complex state functions is a prime candidate for the AmPAL23S8.

The block diagram on the front page shows the basic architecture of this device; a maximum of 23 array inputs and 8 outputs are available. The inputs are connected to a programmable AND array containing 135 product terms (PTs), of which 124 are logical PTs and 11 are control PTs. Before programming, the AND gates are connected to both the true and complement of every input. By selectively programming fuses, the AND gates may be connected to only the true input, the complement input, or to neither type of input, establishing a logical "don't care". When both the true and complement fuses are left intact, a logical FALSE results on the output of the AND gate. An AND gate with all fuses blown will assume the logical TRUE state. The outputs of the AND gates are connected to OR gates.

### Variable Product Term (PT) Distribution

The number of AND gates assigned to each OR gate varies in a fixed manner for each output as shown in the logic diagram (Figure 5). The OR- gate outputs feed dedicated registers and macrocells. Each OR gate averages approximately ten PTs for output registers and macrocells. This gives the capability of using from eight to twelve logical PTs on one output in a single clock cycle (no feedback necessary). Buried state registers (BSRs) have an average of eight PTs per OR gate, providing the capability of using from six to ten logical PTs in one BSR in a single clock cycle.

### Variable Output Architecture: Output Logic Macrocells (OLMs)

An innovation in logic design is the implementation on the AmPAL23S8 of variable output architecture on four of the outputs. These Output Logic Macrocells (OLMs) are user programmable for a great deal of design flexibility. Each of the four OLMs can be independently programmed for eight distinct configurations. The outputs can be either "registered" or "combinatorial;" they can also be individually programmed for active-HIGH or active-LOW polarity. Finally, the feedback paths which feed through the multiplexer back to the AND array can be programmed so that they originate either from the register or from the I/O pin. From the feedback multiplexer both the true and complement of the output going back to the array are available. All possible configurations of the OLMs are illustrated in Figures 4-1 through 4-8. For maximum flexibility, selection of output polarity and feedback path are kept independent of each other.

### Output Registers

In addition to the four OLMs on the AmPAL23S8, there are also four output registers. The data on the output registers may be fed back to the array. When the output is disabled, the pin may be used as an external input. Since each of the eight outputs can obtain feedback from the pins associated with them, all eight of them provide the advantage of being usable dynamically as either inputs or outputs, significantly increasing design flexibility and possibilities.

### Buried State Registers (BSRs)

The six observable Buried State Registers are one of the key features of the AmPAL23S8. All BSR outputs are fed back to the AND array, but they do not use up an output pin. The state of each BSR is, however, observable on an associated output pin by activating the user-programmable Observability product term as well as the appropriate Output Enable product term.

The extensive user-programmable flexibility enhances the usefulness of this device for different types of state machine implementations. The possibility exists to create both the Mealy and Moore type of design in the same device.

### Programmable Output Polarity

Each output has a user-programmable output polarity fuse which, when blown, indicates that the output will be active HIGH, and when intact, active LOW. The obvious benefit of this enhancement is the increased flexibility of design. With the choice of output polarity, there is no need to DeMorganize equations to fit the device, allowing for more efficient designs both in terms of the amount of time spent in design as well as effective utilization of the device.

For further enhancement of the increased logic power of the AmPAL23S8, each output has a PT to control Output Enable (OE) with programmable polarity.

### PRESET/RESET

To improve functionality at the system level, the AmPAL23S8 has additional RESET and PRESET PTs. One PT controls Output Register and BSR PRESET, and one PT controls the RESET for these registers. When the Synchronous PRESET PT is asserted (HIGH), all registers are loaded with a HIGH on the next LOW-to-HIGH clock transition. When the Asynchronous RESET PT is asserted, all registers are immediately loaded with a LOW, independent of the clock. These functions are particularly useful for applications such as system power-up and RESET.

### PRELOAD

In order to simplify testing, the AmPAL23S8 is designed with PRELOAD circuitry that provides an easy method for testing logical functionality. PRELOAD allows any arbitrary "present state" values to be loaded into the OLMs, BSRs and Output Registers of this device. OLM Registers and BSRs are PRELOADed in separate cycles, allowing them to be PRELOADed with different values. Logic verification sequences can be significantly shortened, and all possible state sequences tested, reducing test time and development costs, and guaranteeing proper functionality in system.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. To verify these transitions requires the ability to set the state registers to an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is then clocked into a new state, or "next state," which can be checked to validate the transition from the "present state." In this way, any state transition can be checked.

It is obvious that to attempt the debugging of a design using BSRs without the benefit of PRELOAD capability would be quite difficult. The combination of this feature and the BSRs

being observable is virtually indispensible for efficient and trouble-free state machine design.

### Observability

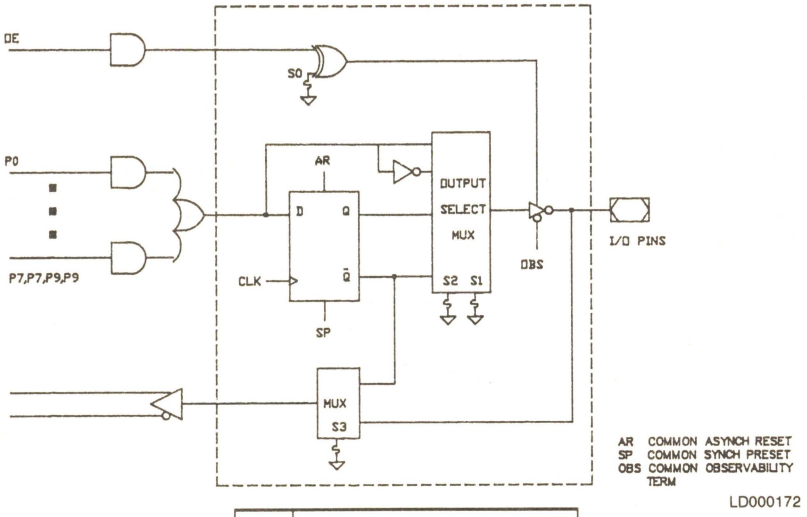
This extra ease of debugging the design comes from the use of the Observability (OBS) PT. When the OBS PT is selected, it disables six of the Output Registers and Macrocell buffers, and enables the BSR buffers onto the output pins associated with them, pins 13 through 18. When the OBS PT is not selected, the Output Registers and Macrocell buffers are enabled and the BSR buffers are disabled. When all the fuses for this PT are intact, (i.e., OBS is not selected), the data from the BSRs will not be visible on the output pins, and the Output Registers and OLMs will be enabled.

### Processing and Fuse Technology

The AmPAL23S8 is manufactured using Advanced Micro Devices' IMOX oxide isolation process. This advanced process permits an increase in density and a decrease in internal capacitance resulting in the fastest possible programmable logic devices.

The AmPAL23S8 is fabricated with AMD's fast programming, highly reliable Platinum-Silicide fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are preprogrammed during manufacturing to ensure extremely high field programming yields ( > 98%), and provide extra test paths to achieve excellent parametric correlation.

Platinum-Silicide was selected as the fuse link material to achieve a well controlled melt rate resulting in large, nonconductive gaps that ensure very stable, long term reliability. Extensive operating testing has proven that this low-field, large-gap technology offers high reliability for fusible link programmable logic.

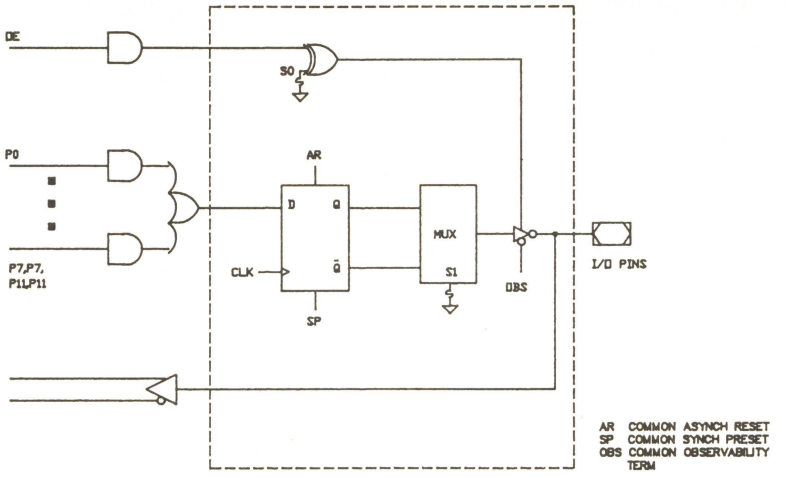



| S0 | OUTPUT ENABLE POLARITY |
|---|---|
| 0 | ENABLED HIGH |
| 1 | ENABLED LOW |

| S1 | S2 | S3 | OUTPUT CONFIGURATION |
|---|---|---|---|
| 0 | 0 | 0 | ACTIVE LOW/REG/REG FEEDBACK |
| 0 | 0 | 1 | ACTIVE LOW/REG/IO FEEDBACK |
| 0 | 1 | 0 | ACTIVE LOW/COMB/REG FEEDBACK |
| 0 | 1 | 1 | ACTIVE LOW/COMB/IO FEEDBACK |
| 1 | 0 | 0 | ACTIVE HIGH/REG/REG FEEDBACK |
| 1 | 0 | 1 | ACTIVE HIGH/REG/IO FEEDBACK |
| 1 | 1 | 0 | ACTIVE HIGH/COMB/REG FEEDBACK |
| 1 | 1 | 1 | ACTIVE HIGH/COMB/IO FEEDBACK |

0 = UNBLOWN FUSE
1 = BLOWN FUSE

**Figure 1. Output Logic Macrocell (OLM)**




| S0 | OUTPUT ENABLE POLARITY |
|---|---|
| 0 | ENABLED HIGH |
| 1 | ENABLED LOW |

| S1 | OUTPUT CONFIGURATION |
|---|---|
| 0 | ACTIVE LOW |
| 1 | ACTIVE HIGH |

0 = UNBLOWN FUSE
1 = BLOWN FUSE

**Figure 2. Output Register With Polarity**

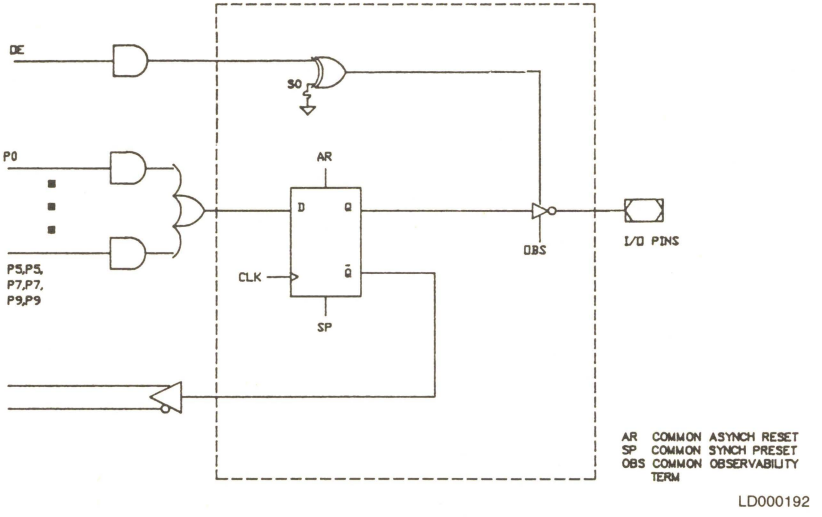
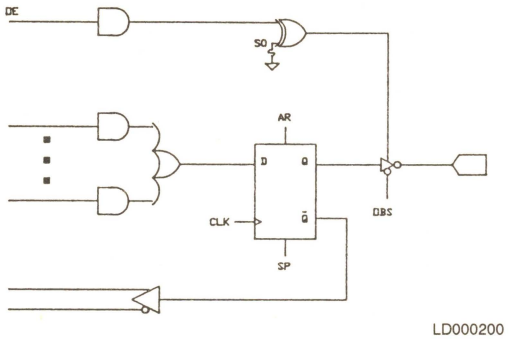


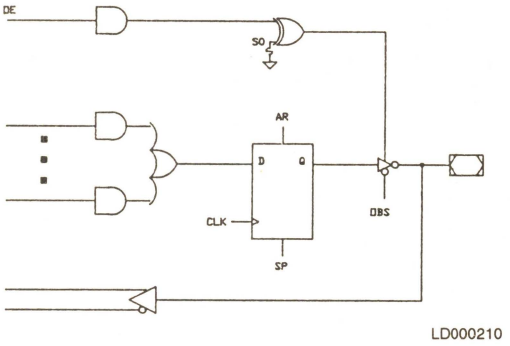

**Figure 3. Buried State Register (BSR)**

## Figure 4. Possible Configurations of the Output Logic Macrocells (OLMs)



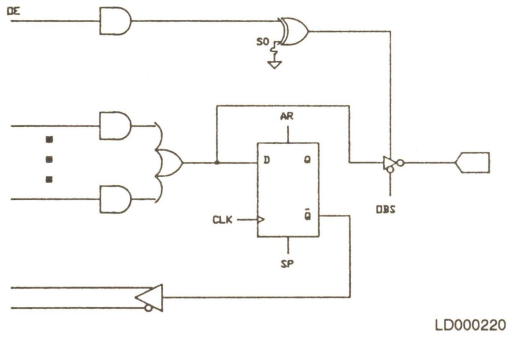

OUTPUT
ACTIVE LOW S1 = 0
REGISTERED S2 = 0
FEEDBACK
REGISTERED S3 = 0

**Figure 4-1.**



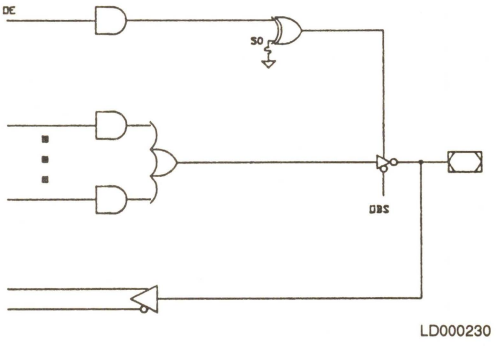

OUTPUT
ACTIVE LOW S1 = 0
REGISTERED S2 = 0
FEEDBACK
I/O PIN S3 = 1

**Figure 4-2.**




OUTPUT
ACTIVE LOW S1 = 0
COMBINATORIAL S2 = 1
FEEDBACK
REGISTERED S3 = 0

**Figure 4-3.**

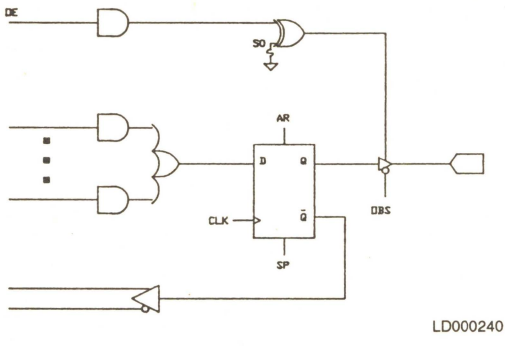



OUTPUT
ACTIVE LOW S1 = 0
COMBINATORIAL S2 = 1
FEEDBACK
I/O PIN S3 = 1

**Figure 4-4.**

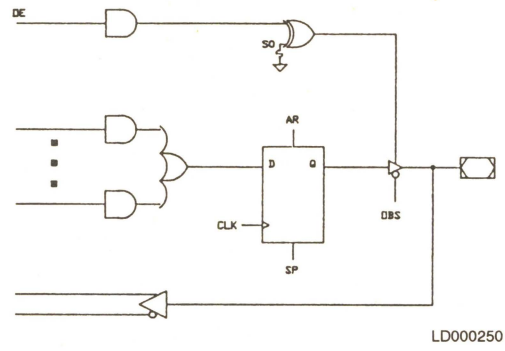
5

**Figure 4. Possible Configurations of the Output Logic Macrocells (OLMs)** (Cont'd.)



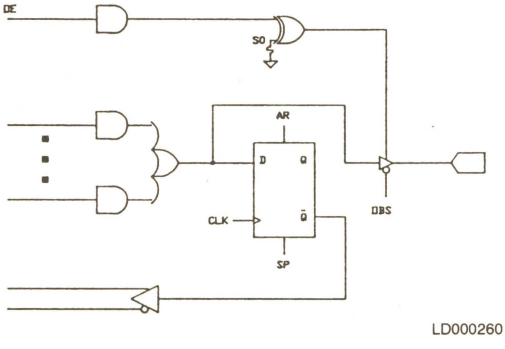

OUTPUT
ACTIVE HIGH S1 = 1
REGISTERED S2 = 0
FEEDBACK
REGISTERED S3 = 0

**Figure 4-5.**



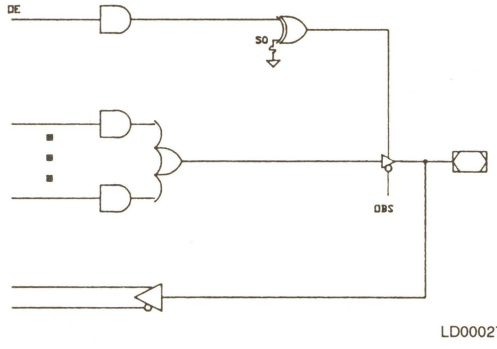

OUTPUT
ACTIVE HIGH S1 = 1
REGISTERED S2 = 0
FEEDBACK
I/O PIN S3 = 1

**Figure 4-6.**




OUTPUT
ACTIVE HIGH S1 = 1
COMBINATORIAL S2 = 1
FEEDBACK
REGISTERED S3 = 0

**Figure 4-7.**




OUTPUT
ACTIVE HIGH S1 = 1
COMBINATORIAL S2 = 1
FEEDBACK
I/O PIN S3 = 1

**Figure 4-8.**

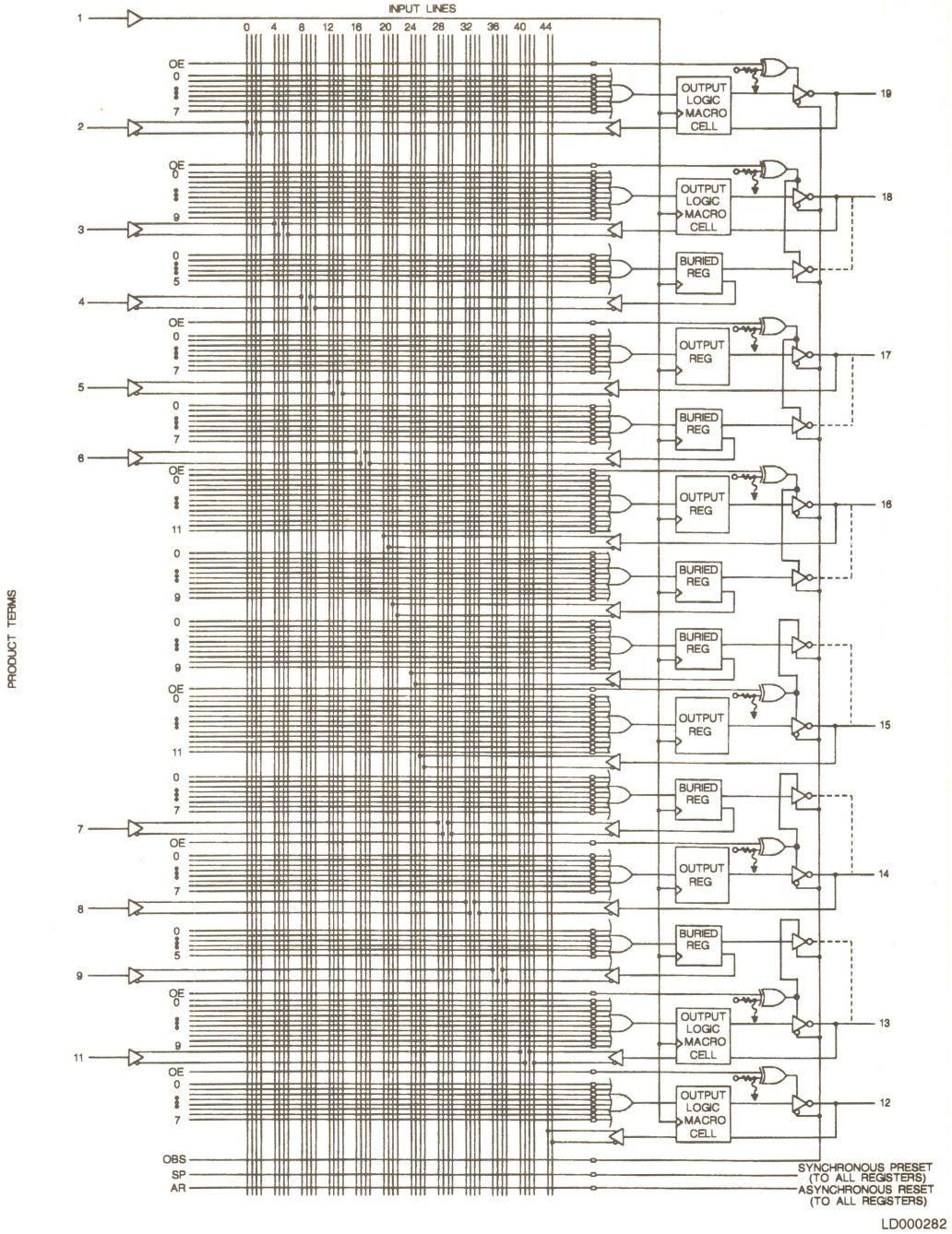
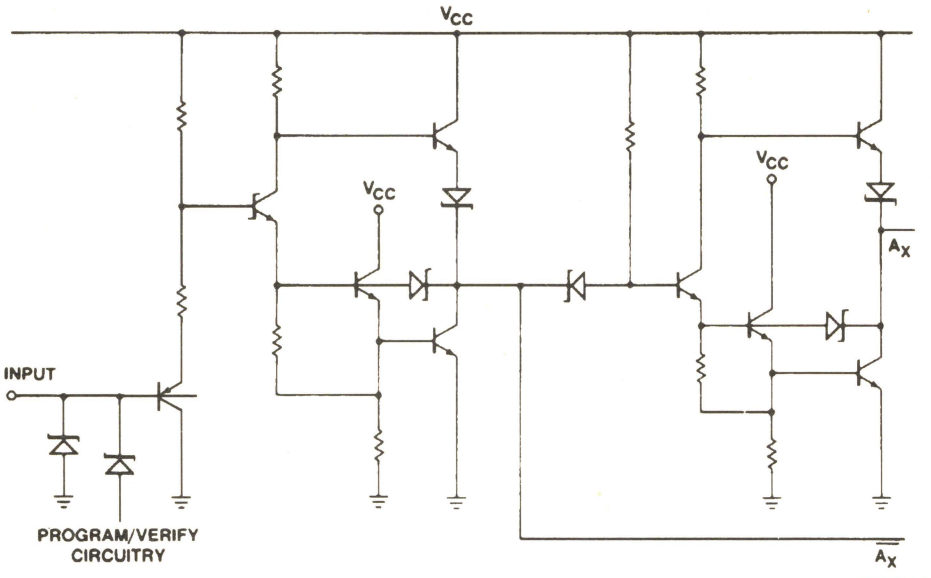



Figure 4. Logic Diagram — AmPAL23S8

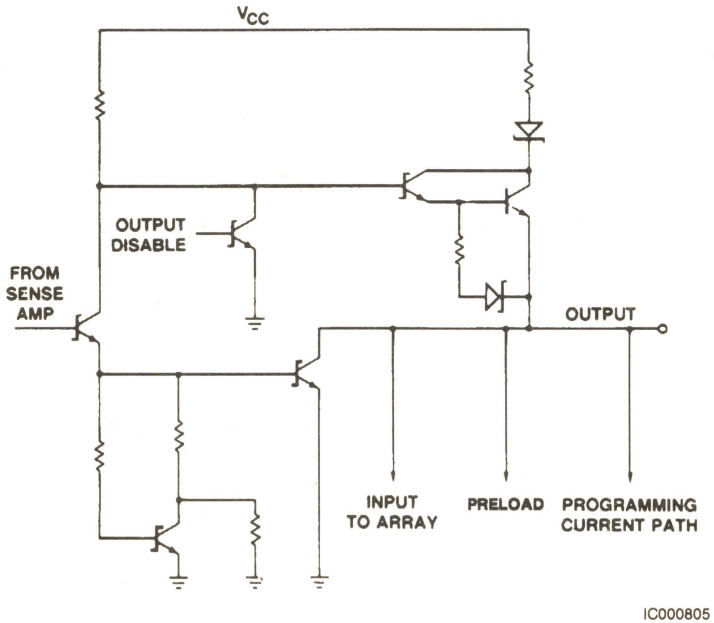
## Input/Output Diagrams

Input




IC000804

Output




IC000805

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Ambient Temperature with
Power Applied ..... +125°C
Supply Voltage to Ground Potential
Continuous (Pin 20 to Pin 10) ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs
(except during programming) ..... −0.5 to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 16 V
Output Current into Outputs
During Programming
(Maximum duration of 1 second) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5.0 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) Operating Free Air ..... 0 to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) Operating Case ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | C Devices | 2.4 | 3.5 | | Volts |
| | | | $I_{OH}$ = −2 mA | E/M Devices | | | | |
| $V_{OL}$ | Output LOW Voltage | | $I_{OL}$ = 16 mA | C Devices | | | 0.50 | Volts |
| | | | $I_{OL}$ = 12 mA | E/M Devices | | | | |
| $V_{IH}$ (Note 2) | Input HIGH level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | 2.0 | | | Volts |
| $V_{IL}$ (Note 2) | Input LOW level | Guaranteed Input Logical LOW Voltage for All Inputs | | | | | 0.8 | Volts |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | −10 | -250 | µA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | 25 | µA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | −30 | −45 | −90 | mA |
| $I_{CC}$ | Power Supply Current | All Inputs = GND, $V_{CC}$ = Max. | | | | | 210 | mA |
| | | | | COMM MIL | | | 215 | mA |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = -18 mA | | | | -0.9 | -1.2 | Volts |
| $I_{OZH}$ $I_{OZL}$ | Output Leakage Current (Note 4) | Vcc = Max., $V_{IN}$ = $V_{IL}$ or $V_{IH}$ | $V_O$ = 2.7 V<br>$V_O$ = 0.4 V | | | | 100<br>100 | µA |

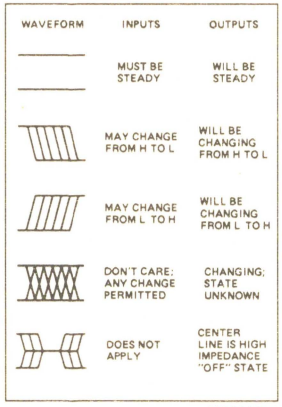
Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).

## Capacitance

| Parameter Symbol | Parameter Description | Test Conditions | | Typ. | Units |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz | Pins 1, 11 | 10 | pF |
| | | | Other Pins | 6 | |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | | 9 | |

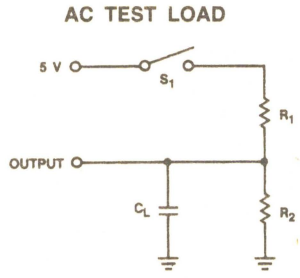
Note: These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

## Key to Switching Waveforms




KS000010

## Switching Test Circuit




TC001190

| TEST OUTPUT LOADS | | | |
|---|---|---|---|
| C Devices | | E/M Devices | |
| $R_1$ | $R_2$ | $R_1$ | $R_2$ |
| 300 | 390 | 390 | 750 |

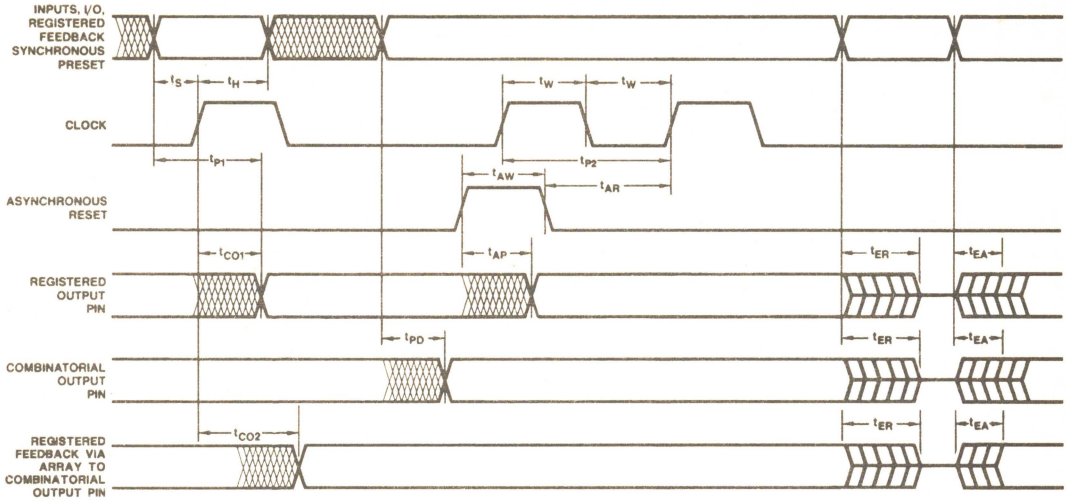
## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 7, 8, 9, 10, 11 tests unless otherwise noted

| No. | Parameter Symbol | Parameter Description | Test Conditions | Typ (Note 1) | C Devices | | | | E/M Devices | | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | -20 | | -25 | | -27 | | -30 | | |
| | | | | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output | C Devices $R_1 = 300$ $R_2 = 390$ | | | 20 | | 25 | | 27 | | 30 | ns |
| 2 | $t_{EA}$ | Input to Output Enable | | | | 25 | | 28 | | 30 | | 33 | ns |
| 3 | $t_{ER}$ | Input to Output Disable | | | | 25 | | 28 | | 30 | | 33 | ns |
| 4 | $t_{CO1}$ | Clock to Output | | | | 13 | | 15 | | 18 | | 20 | ns |
| 5 | $t_{CO2}$ | Reg. Feedback through Array to Combinatorial Output, Relative to External Clock | | | | 28 | | 35 | | 40 | | 45 | ns |
| 6 | $t_S$ | Input or Feedback Setup Time | | | 17 | | 20 | | 22 | | 25 | | ns |
| 7 | $t_H$ | Hold Time | | | 0 | | 0 | | 0 | | 0 | | ns |
| 8 | $t_{P1}$ | Clock Period ($t_S + t_{CO1}$) | | | 30 | | 35 | | 40 | | 45 | | ns |
| 9 | $t_{P2}$ | Minimum Setup Time (Reg. Feedback to Reg. Input Internal Path) | | | 25 | | 30 | | 35 | | 40 | | ns |
| 10 | $t_W$ | Clock Width | | | 12 | | 15 | | 17 | | 20 | | ns |
| 11 | $f_{1MAX}$ | Maximum Frequency ($1/t_{P1}$) | | | | 33 | | 28.5 | | 25 | | 22.5 | MHz |
| 12 | $f_{2MAX}$ | Maximum Frequency ($1/t_{P2}$) | E/M Devices $R_1 = 390$ $R_2 = 750$ | | | 40 | | 33 | | 28.5 | | 25 | MHz |
| 13 | $t_{AW}$ | Asynchronous Reset Width | | | 20 | | 25 | | 27 | | 30 | | ns |
| 14 | $t_{AR}$ | Asynchronous Reset Recovery Time | | | 20 | | 25 | | 27 | | 30 | | ns |
| 15 | $t_{AP}$ | Asynchronous Reset to Registered Output Reset | | | | 25 | | 30 | | 32 | | 35 | ns |

Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high-impedance to HIGH tests and closed for high-impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high-impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high-impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

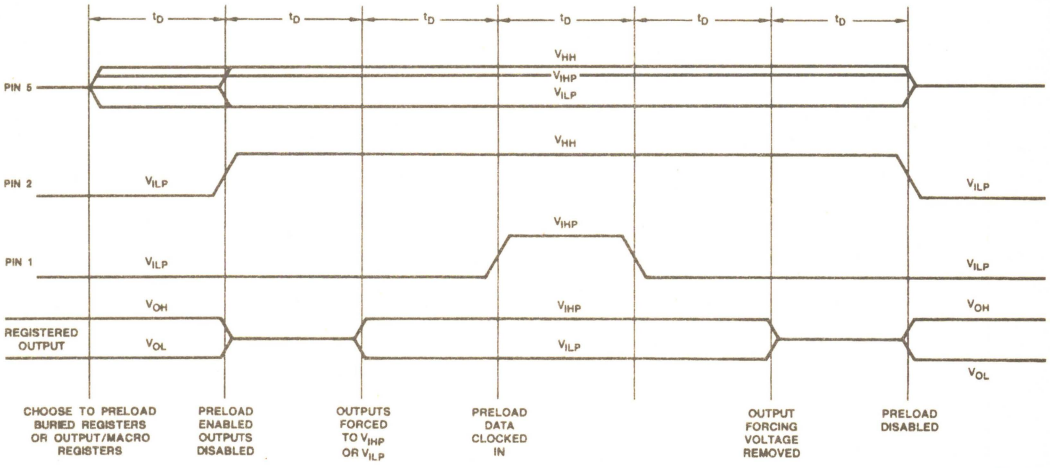
## Switching Waveform




WF022285

## PRELOAD of Output/Macrocell Registers or Buried State Registers

All AmPAL23S8 registers are provided with circuitry to allow loading each register synchronously with a HIGH or LOW. Output/macrocell registers and buried state registers are PRELOADed in separate cycles allowing output/macrocell registers and buried state registers to be PRELOADed with different values. PRELOAD will simplify testing since any state can be loaded into the registers to control testing sequences. The pin levels and timing necessary to perform the PRELOAD function are detailed below.




WF022292

| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |

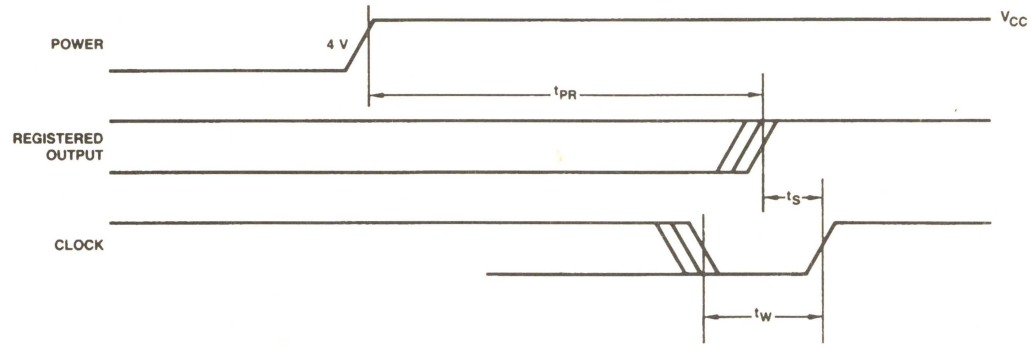
| Register Selection | Pin 5 |
|---|---|
| Buried | $V_{HH}$ |
| Output/Macro | $\leq V_{IHP}$ |

| Level Forced on Register Output Pin During PRELOAD Cycle | Register State After Cycle |
|---|---|
| $V_{IHP}$ | HIGH |
| $V_{ILP}$ | LOW |

## Power-Up Reset

The registered devices in the AMD PAL Family have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to LOW. The output state will depend on the polarity of the output buffer. This feature provides an extra flexibility to the designer and is especially valuable in simplifying state-machine initialization. A timing diagram and parameter table follow. Due to the asynchronous operation of the power-up reset and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to ensure a valid power-up reset. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from LOW to HIGH until all applicable input and feedback setup times are met.




WF022300

| Parameter Symbol | Parameter Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

# AmPAL16R8D Series

## ADVANCE INFORMATION

20-Pin IMOX™ Ultra High-Speed Programmable Array Logic (PAL) Elements

### Distinctive Characteristics

- Superior IMOX technology
  – Guarantees $t_{PD}$ = 10 ns Max. "D" version
- Very high-speed, half-power ("BL") and high-speed, quarter-power ("AQ") versions
- Platinum-silicide fuses and added test words ensure programming yields > 98%
- Post Programming Functional Yields (PPFY) of 99.9%
- PRELOAD feature permits full logical verification
- Reliability assured through more than 70 billion fuse hours of life testing with no failures
- AC and DC parametric testing at the factory through on-board testing circuitry
- Industry-leading quality guarantees

### General Description

AMD PAL devices are high-speed, electrically programmable array logic elements. They utilize the familiar sum-of-products (AND-OR) structure allowing users to program custom logic functions to fit most applications precisely. Typically they are a replacement for low-power Schottky SSI/MSI logic circuits, reducing chip count by more than 5 to 1 and greatly simplifying prototyping and board layout.

Four different devices are available, including both registered and combinatorial devices. This new speed option extends the current AmPAL16R8 Family, allowing the designer to precisely match system requirements.

Please see the "AmPAL16R8 Family" data sheet (Publication No. 03323D) for Block Diagrams.

### Product Selector Guide

**AMD PAL Speed/Power Families**

| Family | $t_S$ (1) ns (Min.) | $t_{PD}$ ns (Max.) | $I_{OL}$ mA (Min.) | $t_{CO}$ ns (Max.) | $I_{CC}$ mA (Max.) |
|---|---|---|---|---|---|
| Ultra High-Speed ("D") Versions | 10 | 10 | 24 | 8 | 180 |

(1) Sequential functions

**AMD PAL Functions**

| Part Number | Array Inputs | Logic | Output Enable | Outputs | Package Pins |
|---|---|---|---|---|---|
| 16R8 | Eight Dedicated, Eight Feedback | Eight 8-Wide AND-OR | Dedicated | Registered Inverting | 20 |
| 16R6 | Eight Dedicated, Six Feedback, Two Bidirectional | Six 8-Wide AND-OR | Dedicated | Registered Inverting | 20 |
| | | Two 7-Wide AND-OR-INVERT | Programmable | Bidirectional | |
| 16R4 | Eight Dedicated, Four Feedback, Four Bidirectional | Four 8-wide AND-OR | Dedicated | Registered Inverting | 20 |
| | | Four 7-Wide AND-OR-INVERT | Programmable | Bidirectional | |
| 16L8 | Ten Dedicated, Six Bidirectional | Eight 7-Wide AND-OR-INVERT | Programmable | Six Bidirectional Two Dedicated | 20 |



08486C/0
JANUARY 1988

# AmPAL16R8 Family

20-Pin IMOX™ Programmable Array Logic (PAL) Elements

## Distinctive Characteristics

- AMD's superior IMOX technology
  - Guarantees $t_{PD}$ = 15 ns Max. "B" Versions
- High-Speed, Half-Power ("AL") and Quarter-Power ("Q") versions
- Platinum-silicide fuses and added test words ensure programming yields > 98%
- Post Programming Functional Yields (PPFY) of 99.9%
- PRELOAD feature permits full logical verification
- Reliability assured through more than 70 billion fuse hours of life testing with no failures
- AC and DC parametric testing at the factory through on-board testing circuitry
- AMD's industry-leading quality guarantees

## General Description

AMD PAL devices are high-speed, electrically programmable array logic elements. They utilize the familiar sum-of-products (AND-OR) structure allowing users to program custom logic functions to fit most applications precisely. Typically they are a replacement for low-power Schottky SSI/MSI logic circuits, reducing chip count by more than 5 to 1 and greatly simplifying prototyping and board layout.

Seven different devices are available, including both registered and combinatorial devices, in six different speed and power versions. The very High-Speed "B" versions ($t_{PD}$ = 15 ns) run approximately 40% faster than the High-Speed "A" versions ($t_{PD}$ = 25 ns). High-Speed, Half-Power "AL" versions ($t_{PD}$ = 25 ns, $I_{CC}$ = 90 mA) are available, as well as Standard-Speed, Half-Power "L" versions ($t_{PD}$ = 35 ns, $I_{CC}$* = 80 mA). Quarter-Power "Q" versions ($t_{PD}$ = 35 ns, $I_{CC}$ = 45 mA) are also available.

Please see the following pages for Block Diagrams.

*Combinatorial functions

## Product Selector Guide

**AMD PAL Speed/Power Families**

| Family | $t_{PD}$ (2) ns (Max.) | | $t_S$ (1) ns (Min.) | | $t_{CO}$ (1) ns (Max.) | | $I_{CC}$ mA (Max.) | $I_{OL}$ mA (Min.) | |
|---|---|---|---|---|---|---|---|---|---|
| | C Devices | M Devices | C Devices | M Devices | C Devices | M Devices | C/M Devices | C Devices | M Devices |
| Very High-Speed ("B") Versions | 15 | 20 | 13 | 18 | 12 | 15 | 180 | 24 | 12 |
| High-Speed ("A") Versions | 25 | 30 | 20 | 25 | 15 | 20 | 180(1)/ 155(2) | 24 | 12 |
| High-Speed, Half-Power ("AL") Versions | 25 | 30 | 20 | 25 | 15 | 20 | 90 | 24 | 12 |
| Standard Versions | 35 | 40 | 30 | 35 | 25 | 25 | 180(1)/ 155(2) | 24 | 12 |
| Half-Power ("L") Versions | 35 | 40 | 30 | 35 | 25 | 25 | 90(1)/ 80(2) | 24 | 12 |
| Quarter-Power ("Q") Versions | 35 | 40 | 30 | 35 | 25 | 25 | 45 | 12 | 8 |

(1) Sequential functions
(2) Combinatorial functions

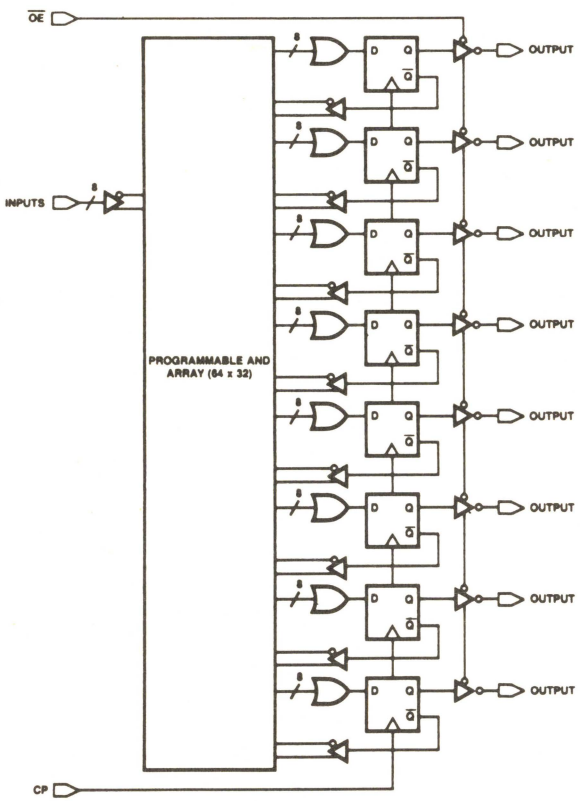
**AMD PAL FUNCTIONS**

| Part Number | Array Inputs | Logic | Output Enable | Outputs | Package Pins |
|---|---|---|---|---|---|
| 16R8 | Eight Dedicated, Eight Feedback | Eight 8-Wide AND-OR | Dedicated | Registered Inverting | 20 |
| 16R6 | Eight Dedicated, Six Feedback, Two Bidirectional | Six 8-Wide AND-OR | Dedicated | Registered Inverting | 20 |
| | | Two 7-Wide AND-OR-INVERT | Programmable | Bidirectional | |
| 16R4 | Eight Dedicated, Four Feedback, Four Bidirectional | Four 8-wide AND-OR | Dedicated | Registered Inverting | 20 |
| | | Four 7-Wide AND-OR-INVERT | Programmable | Bidirectional | |
| 16L8 | Ten Dedicated, Six Bidirectional | Eight 7-Wide AND-OR-INVERT | Programmable | Six Bidirectional Two Dedicated | 20 |

03323F/0
JANUARY 1988

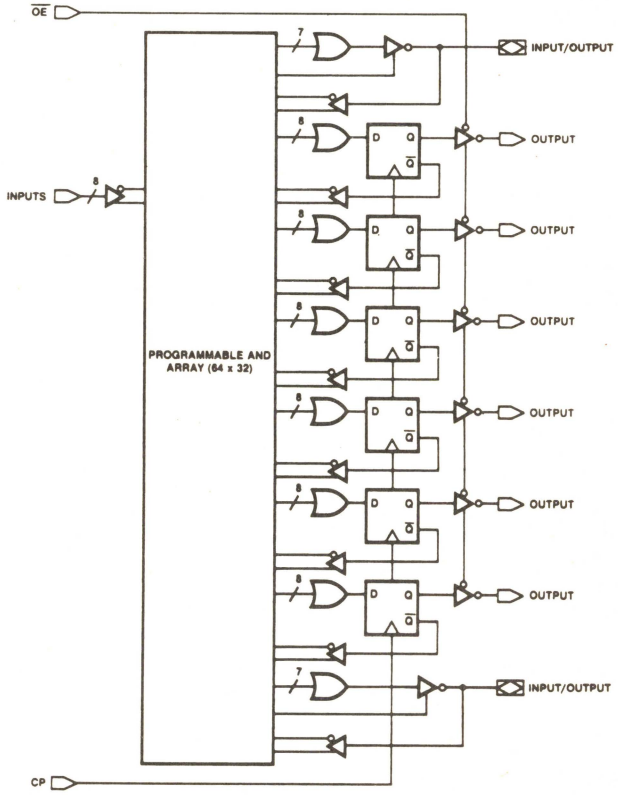
## Block Diagrams

### AmPAL16R8




BD002720

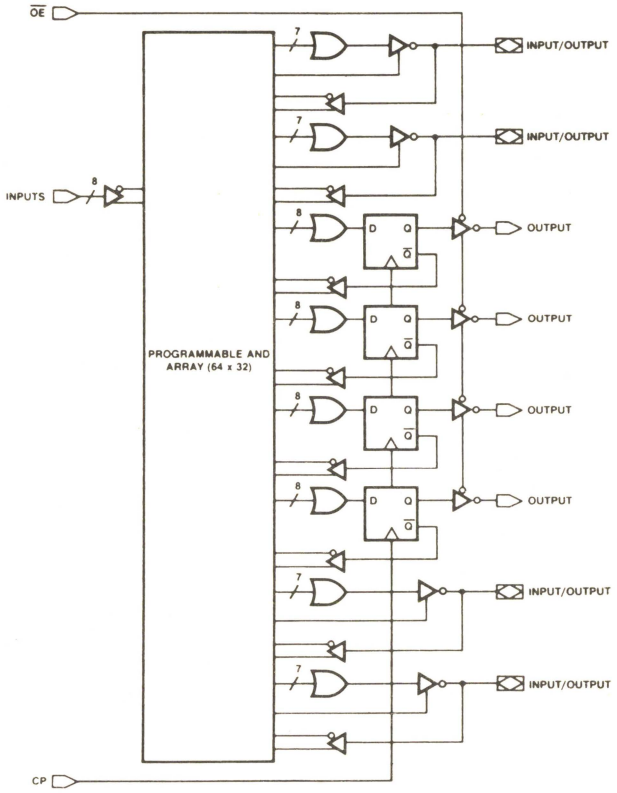
### AmPAL16R6




BD002730

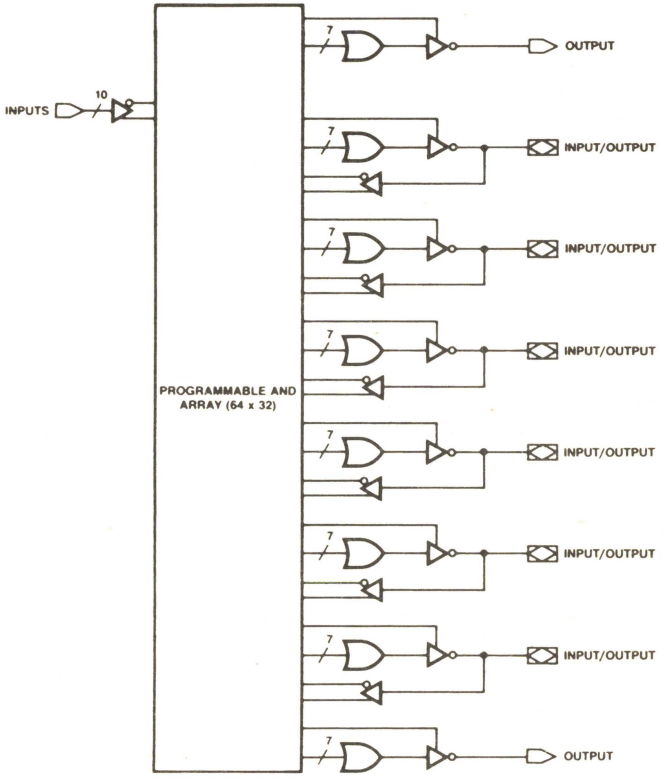
## Block Diagrams (Cont'd.)

### AmPAL16R4




BD002740

### AmPAL16L8

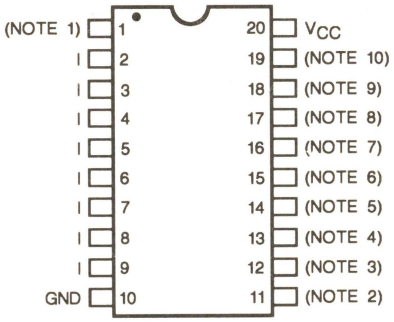



BD002750

## Connection Diagrams

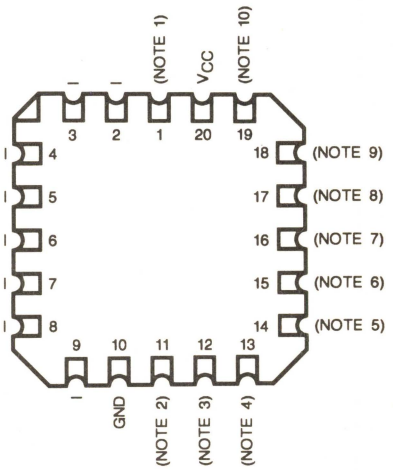
Top View

DIPs*




CD010020

LCC**




CD010030

Note: Pin 1 is marked for orientation.

Notes:

| | 16L8 | 16R8 | 16R6 | 16R4 |
|---|---|---|---|---|
| 1 | I | CLK | CLK | CLK |
| 2 | I | OE | OE | OE |
| 3 | O | O | I/O | I/O |
| 4 | I/O | O | O | I/O |
| 5 | I/O | O | O | O |
| 6 | I/O | O | O | O |
| 7 | I/O | O | O | O |
| 8 | I/O | O | O | O |
| 9 | I/O | O | O | I/O |
| 10 | O | O | I/O | I/O |

*Also available in 20-Pin Ceramic Flatpack. Pinouts identical to DIPs.
**Also available in 20-Pin Plastic Leaded Chip Carrier. Pinouts identical to LCC.

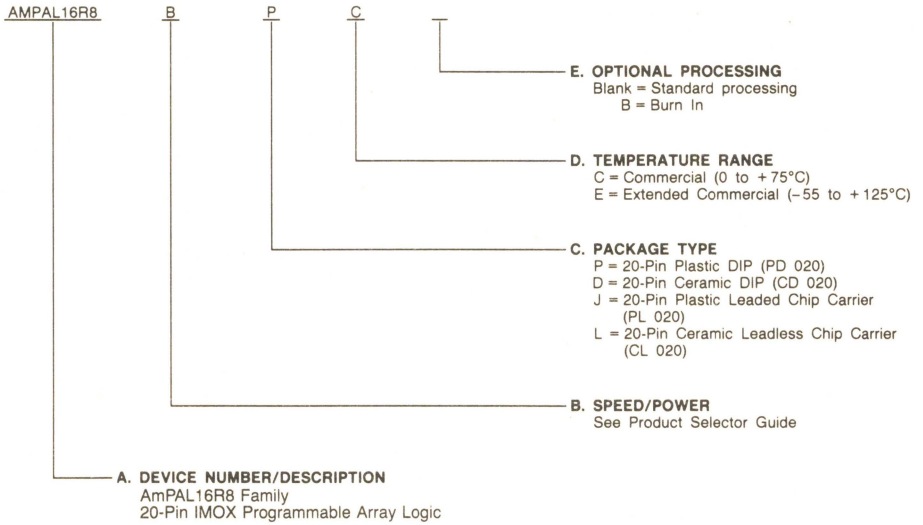
## Pin Designations

I = Input
I/O = Input/Output
O = Output
$V_{CC}$ = Supply Voltage
GND = Ground
CLK = Clock
OE = Output Enable

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AMPAL16R8/B/A/AL/L/Q | PC, DC, DCB, DE, JC, LC, LE |
| AMPAL16R6/B/A/AL/L/Q | |
| AMPAL16R4/B/A/AL/L/Q | |
| AMPAL16L8/B/A/AL/L/Q | |

**Valid Combinations**

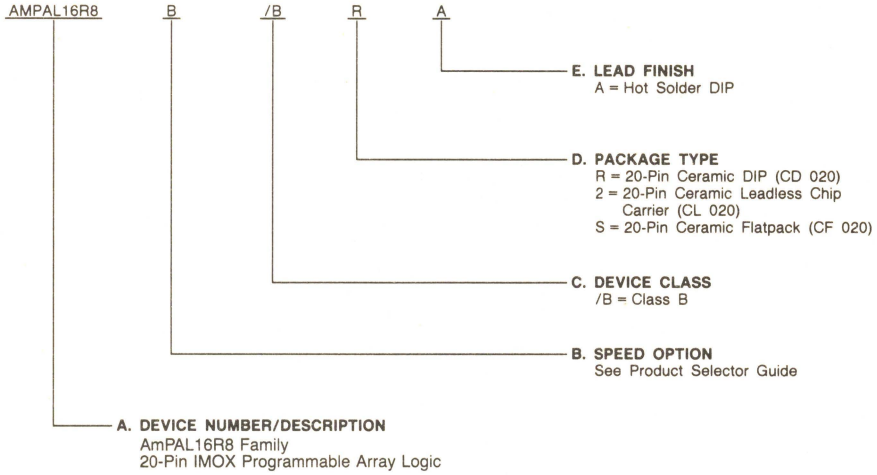
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information (Cont'd.)

### APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. CPL (Controlled Products List) products are processed in accordance with MIL-STD-883C, but are inherently non-compliant because of package, solderability, or surface treatment exceptions to those specifications. The order number (Valid Combination) for APL products is formed by a combination of: **A. Device Number**

**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL16R8/B/A/AL/L/Q | /BRA, /B2A, /BSA |
| AMPAL16R6/B/A/AL/L/Q | |
| AMPAL16R4/B/A/AL/L/Q | |
| AMPAL16L8/B/A/AL/L/Q | |

#### Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

#### Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 4, 9, 10, 11.

5

## DESC Certified PAL Devices

| Generic | AMD Part Number | DESC Numbers |
|---|---|---|
| 16L8 | AmPAL16L8A/BRA | 8103607RX |
| | AmPAL16L8A/B2A | 81036072X |
| | AmPAL16L8A/BSA | 8103607SX |
| | AmPAL16L8L/BRA | 8103611RX |
| | AmPAL16L8L/B2A | 81036112X |
| | AmPAL16L8L/BSA | 8103611SX |
| | AmPAL16L8/BRA | 8103601RX |
| | AmPAL16L8/B2A | 81036012X |
| 16R8 | AmPAL16R8A/BRA | 8103608RX |
| | AmPAL16R8A/B2A | 81036082X |
| | AmPAL16R8A/BSA | 8103608SX |
| | AmPAL16R8L/BRA | 8103612RX |
| | AmPAL16R8L/B2A | 81036122X |
| | AmPAL16R8L/BSA | 8103612SX |
| | AmPAL16R8/BRA | 8103602RX |
| | AmPAL16R8/B2A | 81036022X |
| 16R6 | AmPAL16R6A/BRA | 8103609RX |
| | AmPAL16R6A/B2A | 81036092X |
| | AmPAL16R6A/BSA | 8103609SX |
| | AmPAL16R6L/BRA | 8103613RX |
| | AmPAL16R6L/B2A | 81036132X |
| | AmPAL16R6L/BSA | 8103613SX |
| | AmPAL16R6/BRA | 8103603RX |
| | AmPAL16R6/B2A | 81036032X |
| 16R4 | AmPAL16R4A/BRA | 8103610RX |
| | AmPAL16R4A/B2A | 81036102X |
| | AmPAL16R4A/BSA | 8103610SX |
| | AmPAL16R4L/BRA | 8103614RX |
| | AmPAL16R4L/B2A | 81036142X |
| | AmPAL16R4L/BSA | 8103614SX |
| | AmPAL16R4/BRA | 8103604RX |
| | AmPAL16R4/B2A | 81036042X |

## Functional Description

### AMD PAL Family Characteristics

All members of the AMD PAL Family have common electrical characteristics and programming procedures. All parts are produced with a fusible link at each input to the AND-gate array, and connections may be selectively removed by applying appropriate voltages to the circuit.

Initially the AND gates are connected, via fuses, to both the TRUE and complement of each input. By selective programming of fuses the AND gates may be "connected" to only the TRUE input (by blowing the complement fuse), to only the complement input (by blowing the TRUE fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the TRUE and complement fuses are left intact a logical FALSE results on the output of the AND gate, while all fuses blown results in a logical-TRUE state. The outputs of the AND gates are connected to fixed-OR gates. The only limitations imposed are the number of inputs to the AND gates (up to 16) and the number of AND gates per OR (up to 8).

All parts are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are pre-programmed during manufacturing to ensure extremely high field programming yields ( > 98%), and provide extra test paths to achieve excellent parametric correlation.

### Power-Up RESET

The registered devices in the AMD PAL family have been designed to reset during system power-up. Following power-up, all registers will be initialized to zero, setting all the outputs to a logic 1. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

### PRELOAD

AMD PAL devices are designed with unique PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the registered output of an AMD PAL device.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

## Security Fuse

An additional fuse is provided on each AMD PAL circuit to prevent unauthorized copying of AMD PAL device fuse patterns when design security is desired. Blowing the security fuse blocks entry to the fuse pattern verify mode.

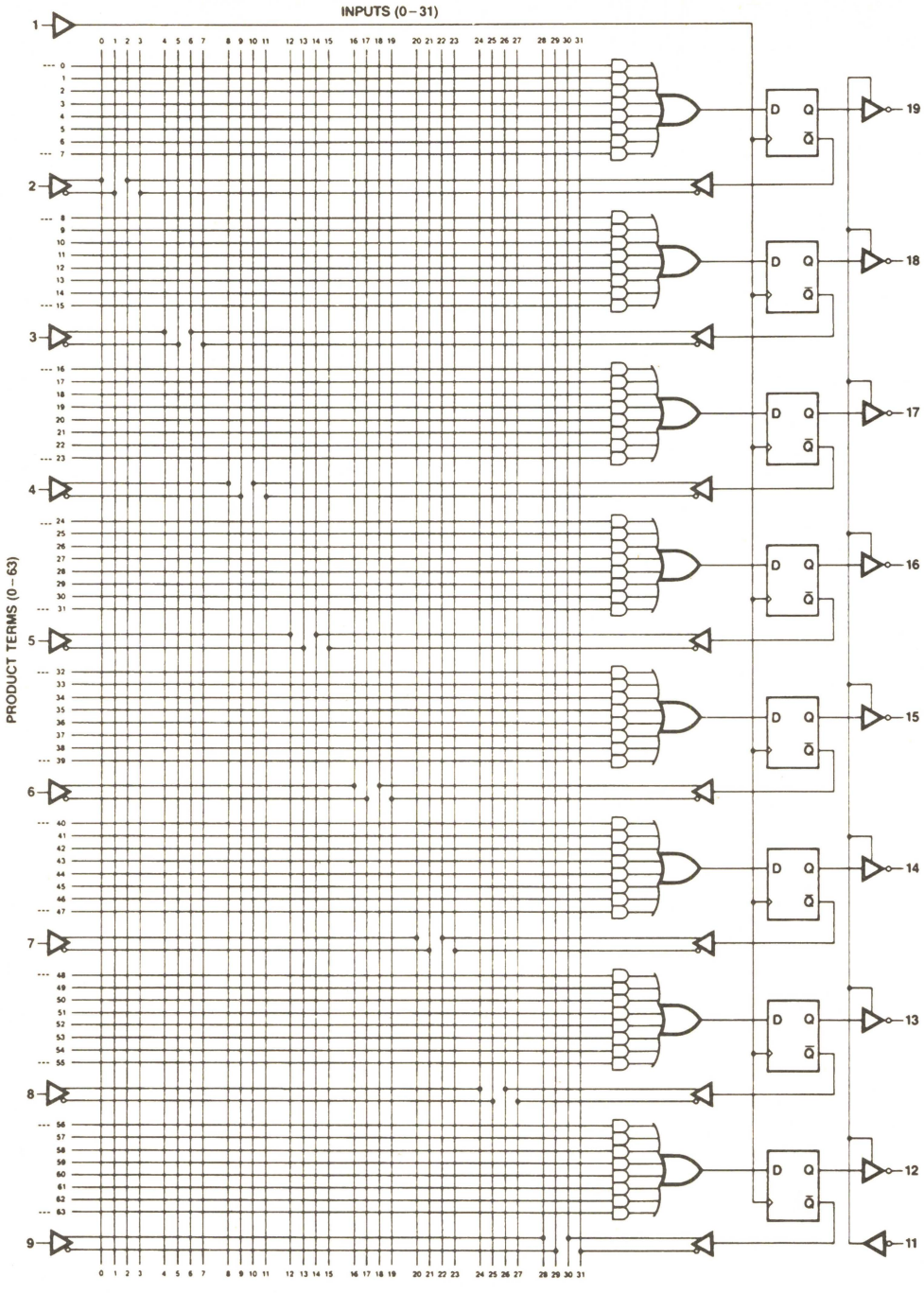



Figure 1. AmPAL16R8 Logic Diagram

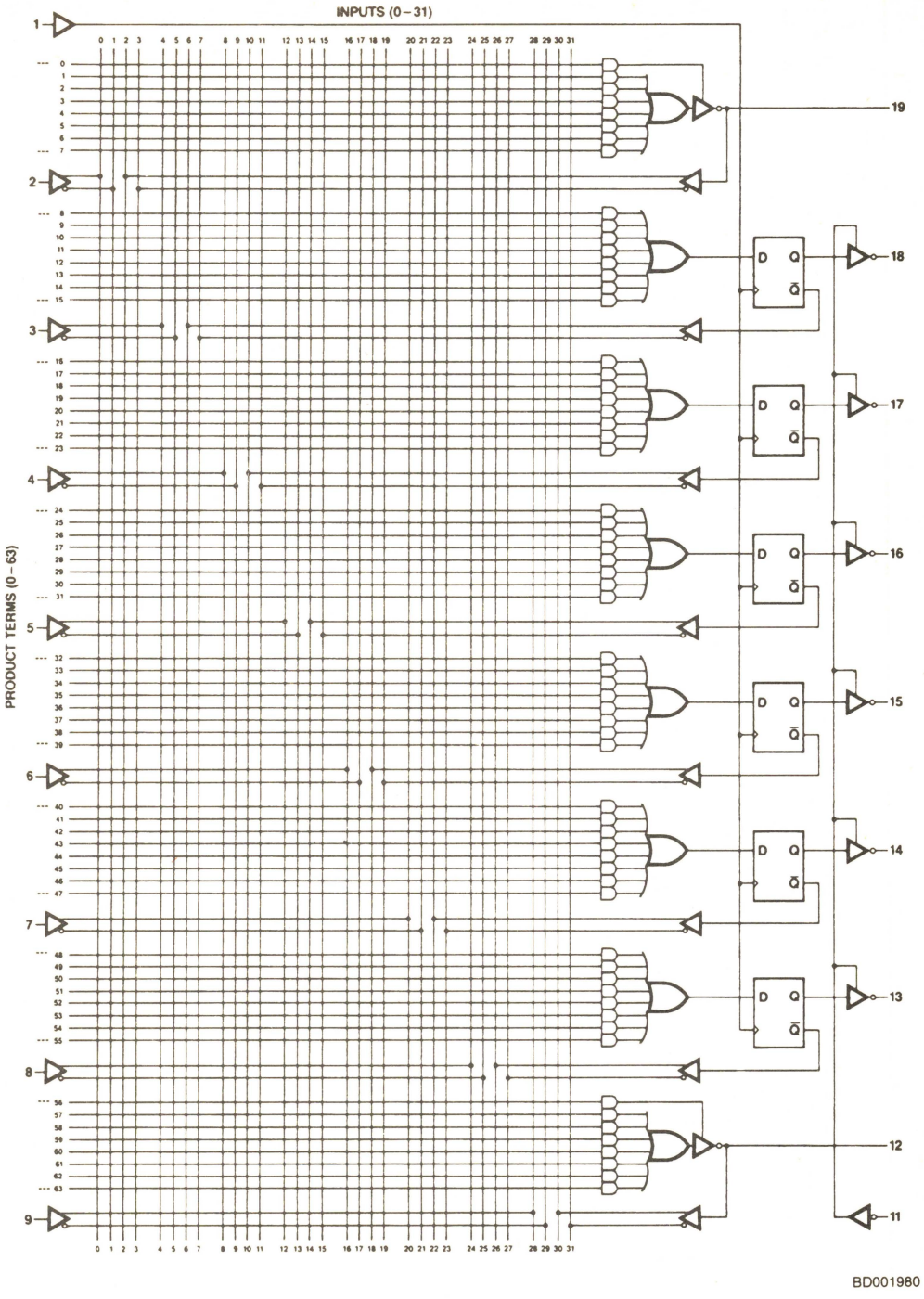



Figure 2. AmPAL16R6 Logic Diagram

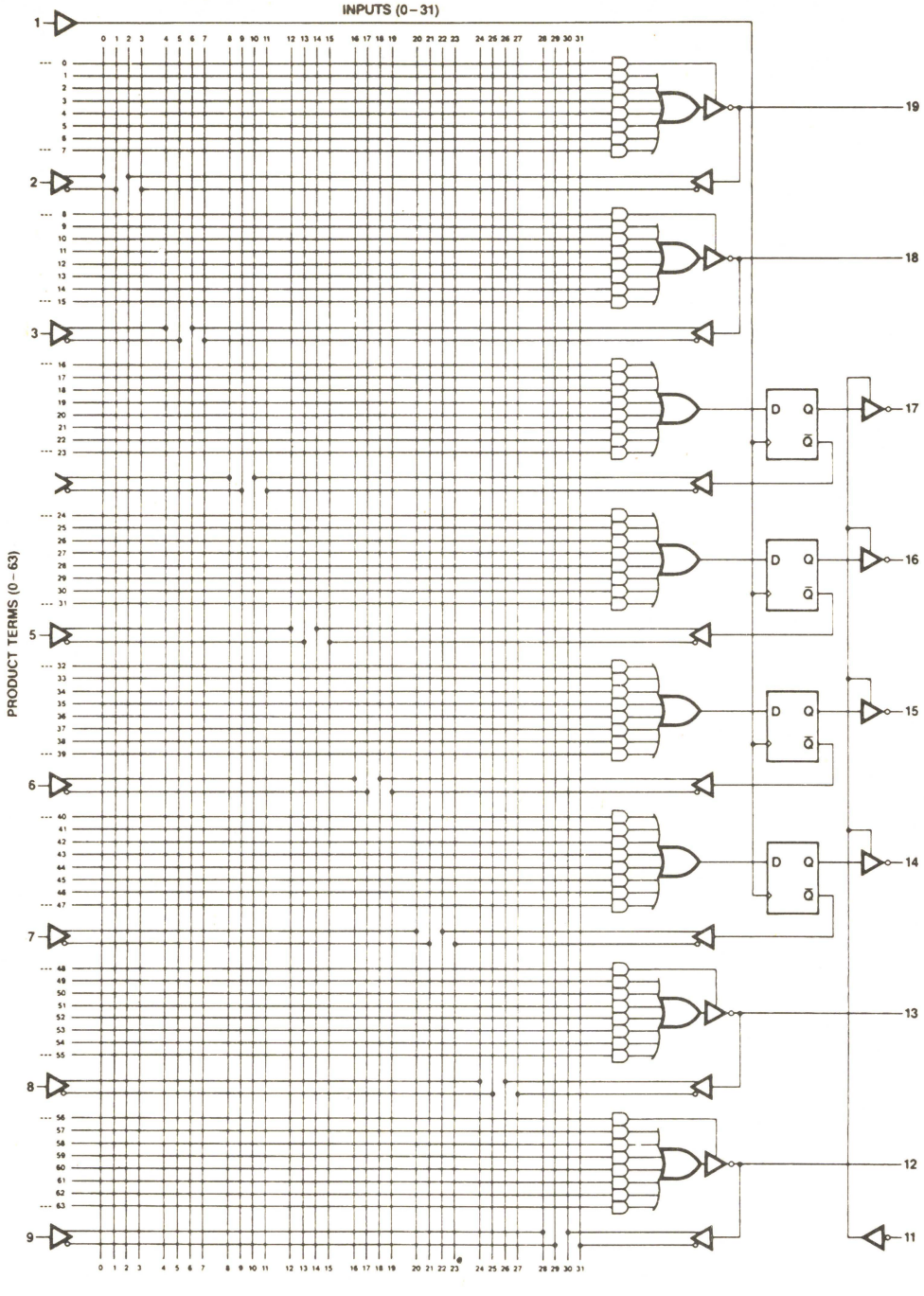



Figure 3. AmPAL16R4 Logic Diagram

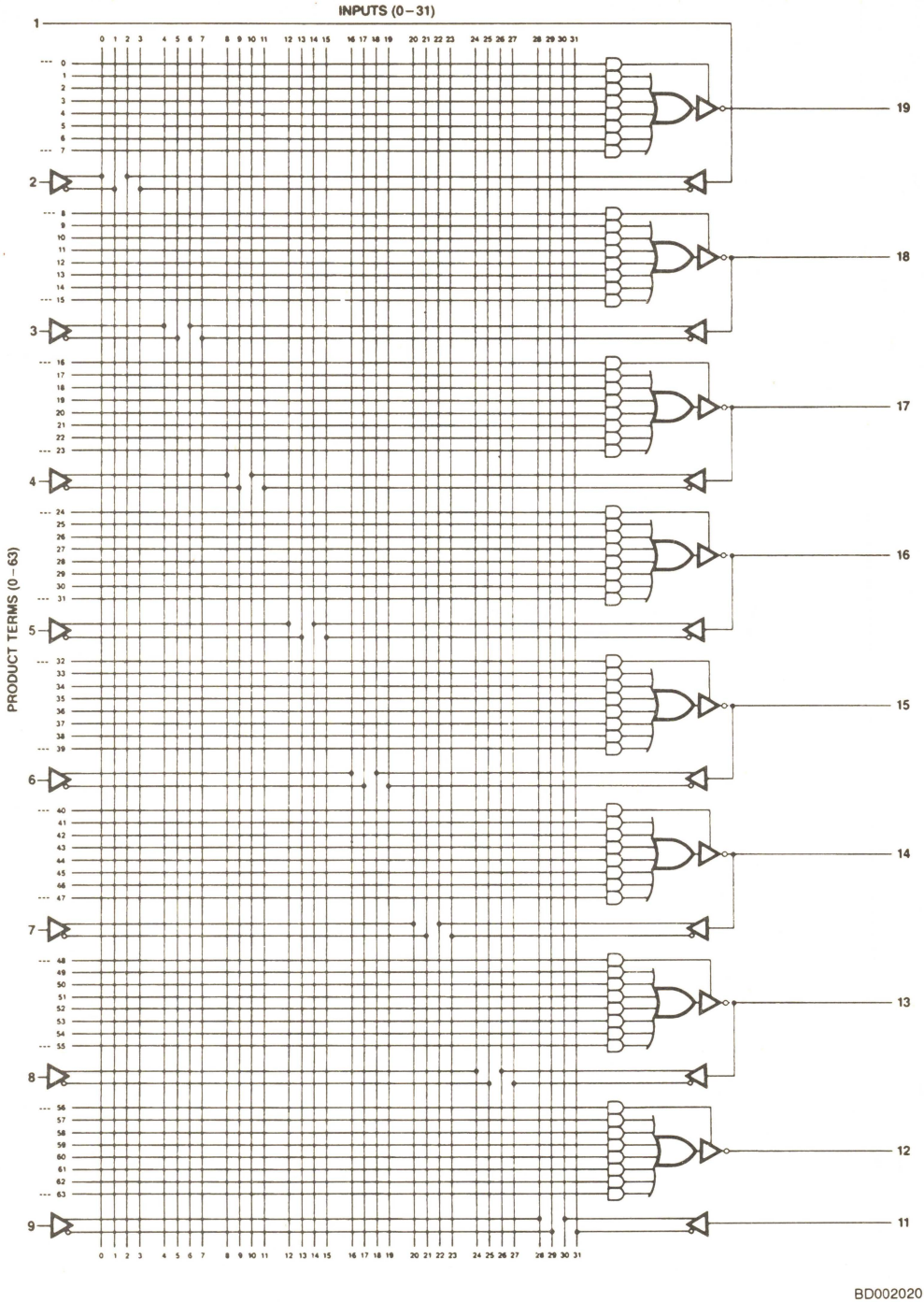


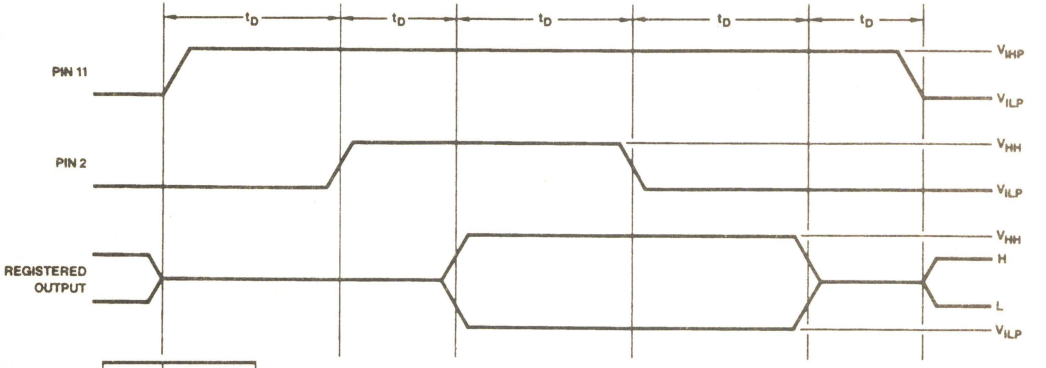

Figure 4. AmPAL16L8 Logic Diagram

## Applications

### PRELOAD of Registered Outputs

AMD PAL registered outputs are designed with extra circuitry to allow loading each register asynchronously to either a HIGH or LOW state. This feature simplifies testing since any initial state for the registers can be set to optimize test sequencing.

The pin levels and timing necessary to perform the PRELOAD function are detailed below:




PF001141

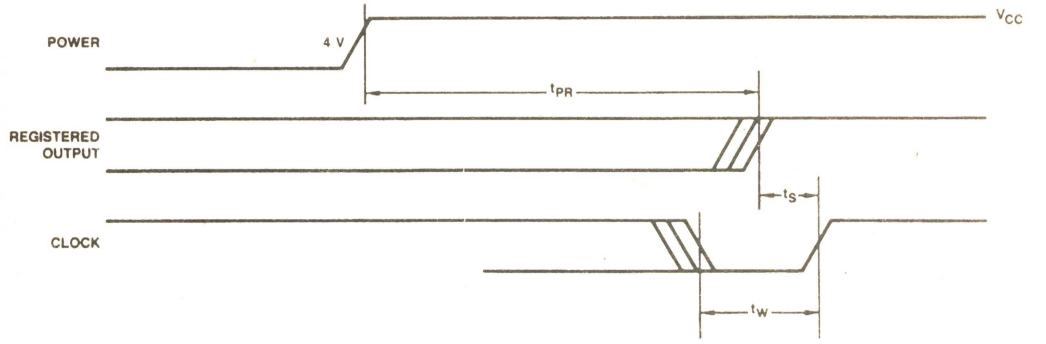
| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |
| $V_{CCH}$ | 5.4 | 6.0 |

| Level forced on registered output pin during PRELOAD cycle | State of the output pin after cycle |
|---|---|
| $V_{HH}$ | HIGH |
| 0 V to $V_{CCH}$ or OPEN | LOW |

### Power-Up Reset

The registered devices in the AMD PAL Family have been designed to reset during system power-up. Due to the asynchronous operation of the power-up reset and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to ensure a valid power-up reset. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from low to high until all applicable input and feedback setup times are met.




WF022300

| Parameters | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
(Pin 20 to Pin 10) Continuous ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs
(Except During Programming) .....−0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 21 V
Output Current Into Outputs During
Programming (Max Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) ..... 0 to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

## DC Characteristics over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3, 4 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | "Q" | $I_{OH}$ = −2 mA | COM'L | 2.4 | 3.5 | | V |
| | | | All others | $I_{OH}$ = −3.2 mA | COM'L | | | | |
| | | | | $I_{OH}$ = −2 mA | MIL | | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | "B," "A," "Std." "AL," & "L" | $I_{OL}$ = 24 mA | COM'L | | | 0.5 | V |
| | | | | $I_{OL}$ = 12 mA | MIL | | | | |
| | | | "Q" | $I_{OL}$ = 12 mA | COM'L | | | | |
| | | | | $I_{OL}$ = 8 mA | MIL | | | | |
| $V_{IH}$ (Note 2) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | | 2.0 | | 5.5 | V |
| $V_{IL}$ (Note 2) | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs | | | | | | 0.8 | V |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $I_{IN}$ = 0.40 V | | "B," "AL," & "Q" | | | −20 | −100 | μA |
| | | | | "A," "L," & "Std." | | | −20 | −250 | |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | | 25 | μA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | | −30 | −60 | −90 | mA |
| $I_{CC}$ | Power Supply Current | All Inputs = GND, $V_{CC}$ = Max. | | 16L8A, 16L8, | | | 110 | 155 | mA |
| | | | | 16L8L, | | | 55 | 80 | |
| | | | | 16R8B, 16R6B, 16R4B, 16L8B, 16R8A, 16R6A, 16R4A, 16R8, 16R6, 16R4 | | | | 180 | |
| | | | | 16R8L, 16R6L, 16R4L, 16L8AL, 16R8AL, 16R6AL, 16R4AL | | | 60 | 90 | |
| | | | | 16L8Q, 16R8Q, 16R6Q, 16R4Q | | | | 45 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | | −0.9 | −1.2 | V |
| $I_{OZH}$ | Output Leakage Current (Note 4) | Vcc = Max., $V_{IN}$ = $V_{IL}$ or $V_{IH}$ | | $V_O$ = 2.7 V | | | | 100 | μA |
| $I_{OZL}$ | | | | $V_O$ = 0.4 V | | | | −100 | |
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz (Note 5) | | | | | 6 | | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz (Note 5) | | | | | 9 | | |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).
5. These parameters are not 100% tested, but are periodically sampled.

5

## Switching Characteristics over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

### Commercial Range

| No. | Parameter Symbol | Parameter Description | "B" Version | | | "A" & "AL" Version | | | "Std," "L" & "Q" Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 16L8, 16R6, 16R4, | | 12 | 15 | | 17 | 25 | | 23 | 35 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 16L8, 16R6, 16R4, 16H8 | | 12 | 15 | | 17 | 25 | | 23 | 35 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 16L8, 16R6, 16R4, 16H8 | | 12 | 15 | | 17 | 25 | | 23 | 35 | ns |
| 4 | $t_{PZX}$ | Pin 11 to Output Enable 16R8, 16R6, 16R4 | | 8 | 15 | | 12 | 20 | | 17 | 25 | ns |
| 5 | $t_{PXZ}$ | Pin 11 to Output Disable 16R8, 16R6, 16R4 | | 8 | 15 | | 12 | 20 | | 17 | 25 | ns |
| 6 | $t_{CO}$ | Clock to Output 16R8, 16R6, 16R4 | | 8 | 12 | | 12 | 15 | | 17 | 25 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 16R8, 16R6, 16R4 | 13 | 10 | | 20 | 15 | | 30 | 20 | | ns |
| 8 | $t_H$ | Hold Time 16R8, 16R6, 16R4 | 0 | −8 | | 0 | −10 | | 0 | −10 | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 25 | | | 35 | | | 55 | | | ns |
| 10 | $t_W$ | Clock Width | 10 | | | 15 | | | 25 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 40 | | | 28.5 | | | 18 | MHz |

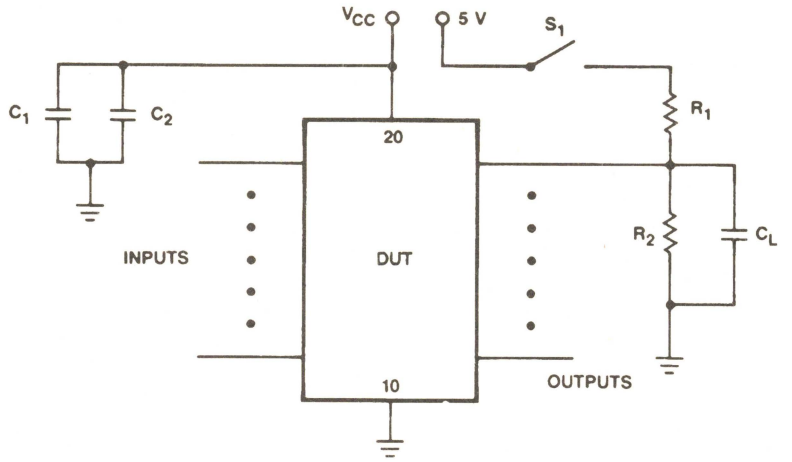
Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

### Military Range

| No. | Parameter Symbol | Parameter Description | "B" Version | | | "A" & "AL" Version | | | "Std," "L" & "Q" Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 16L8, 16R6, 16R4, | | 12 | 20 | | 17 | 30 | | 23 | 40 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 16L8, 16R6, 16R4, 16H8 | | 12 | 20 | | 17 | 30 | | 23 | 40 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 16L8, 16R6, 16R4, 16H8 | | 12 | 20 | | 17 | 30 | | 23 | 40 | ns |
| 4 | $t_{PZX}$ | Pin 11 to Output Enable 16R8, 16R6, 16R4 | | 8 | 20 | | 12 | 25 | STD / L | 17 | 25 / 30 | ns |
| | | | | | | | | | Q | | | |
| 5 | $t_{PXZ}$ | Pin 11 to Output Disable 16R8, 16R6, 16R4 | | 8 | 20 | | 12 | 25 | STD / L | 17 | 25 / 30 | ns |
| | | | | | | | | | Q | | | |
| 6 | $t_{CO}$ | Clock to Output 16R8, 16R6, 16R4 | | 8 | 15 | | 12 | 20 | | 17 | 25 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 16R8, 16R6, 16R4 | 18 | 10 | | 25 | 15 | | 35 | 20 | | ns |
| 8 | $t_H$ | Hold Time 16R8, 16R6, 16R4 | 0 | −8 | | 0 | −10 | | 0 | −10 | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 33 | | | 45 | | | 60 | | | ns |
| 10 | $t_W$ | Clock Width | 12 | | | 20 | | | 25 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 30 | | | 22 | | | 16.5 | MHz |

Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

## Switching Test Circuit



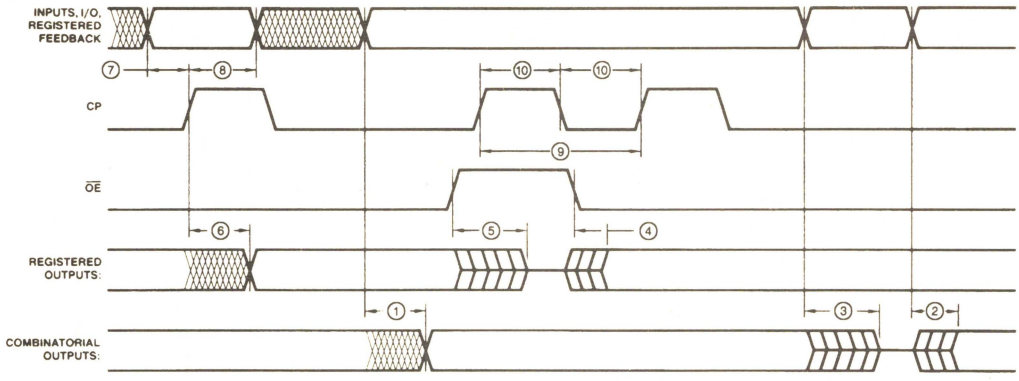

TC003050

Note: $C_1$ and $C_2$ are to bypass $V_{CC}$ to ground.

| TEST OUTPUT LOADS | | | | |
|---|---|---|---|---|
| | "Std," "B," "A," "AL" & "L" | | "Q" | |
| Name | Commercial | Military | Commercial | Military |
| $R_1$ | 200 Ω | 390 Ω | 390 Ω | 600 Ω |
| $R_2$ | 390 Ω | 750 Ω | 750 Ω | 1200 Ω |
| $C_1$ | 1 μF | 1 μF | 1 μF | 1 μF |
| $C_2$ | 0.1 μF | 0.1 μF | 0.1 μF | 0.1 μF |
| $C_L$ | 50 pF | 50 pF | 50 pF | 50 pF |

5

## Switching Waveforms



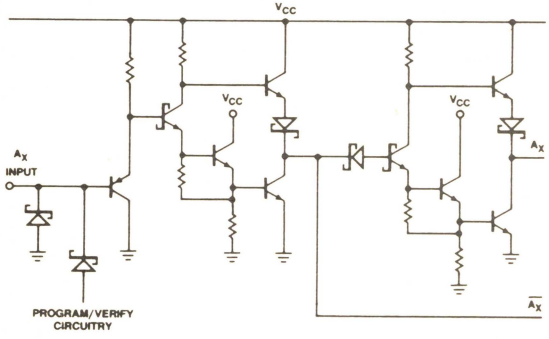

WF002571

## Key to Timing Diagram

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | MUST BE STEADY | WILL BE STEADY |
| | MAY CHANGE FROM H TO L | WILL BE CHANGING FROM H TO L |
| | MAY CHANGE FROM L TO H | WILL BE CHANGING FROM L TO H |
| | DON'T CARE; ANY CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | DOES NOT APPLY | CENTER LINE IS HIGH IMPEDANCE "OFF" STATE |

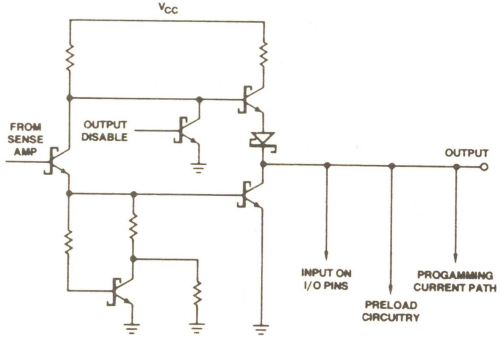
KS000010

## Input Circuitry




IC000720

## Output Circuitry




IC000730

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

# AmPAL18P8

20-Pin IMOX™ Programmable Array Logic

## Distinctive Characteristics

- Individually programmable output polarity on each output
- Pin compatible superset of most combinatorial 20-pin PAL devices
- Eight logical product terms per output for increased logic power
- Increased input/output flexibility
  - 18 possible array inputs
  - Eight bidirectional I/Os with individually controllable output enable
- Ultra high-speed version $t_{PD}$ = 15 ns maximum
- Superior quality
  - AC and DC parametric testing performed on every part
  - Extensive on-chip test circuitry ensures post-programming functional yield (PPFY) of 99.9%
- Platinum-Silicide fuses ensure high programming yield > 98%, fast programming and unsurpassed reliability
- Replaces 13 combinatorial 20-Pin PAL devices

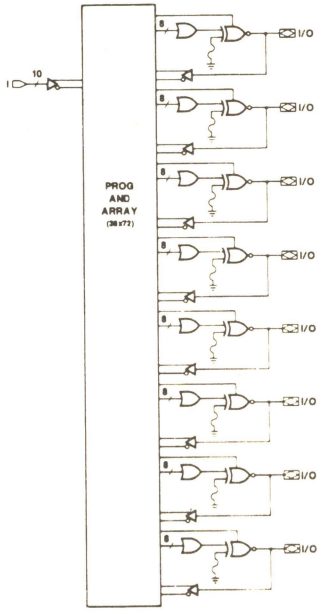
## General Description

The AmPAL18P8 is an ultra high-performance, functionally enhanced 20-pin Programmable Array Logic element. It utilizes the familiar sum-of-products (AND-OR) structure allowing users to program custom logic functions to precisely fit their application.

The AmPAL18P8 offers significantly enhanced functional capabilities when compared to other combinatorial 20-pin PAL devices. These include two additional bidirectional I/O pins as well as additional product terms (bringing each output to eight logical and one three-state control product term) for extra logic power. The device also features individually user programmable output polarity, giving the designer the capability to handle both active HIGH and active LOW outputs on the same device.

A wide variety of speed/power selections is available, allowing precise matching to system requirements. The ultra high-speed version offers 15 ns maximum input to output propagation delay, opening up many new applications for the use of programmable logic.

## Block Diagram




BD005942

### PRODUCT SELECTOR GUIDE

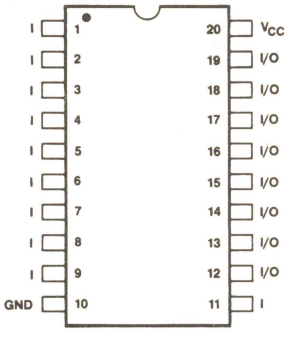
| Family Part No. | AmPAL18P8 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Power Grade | Quarter Power | | Half Power | | | | Full Power | | | |
| Ordering Part No. | 18P8Q | | 18P8L | | 18P8AL | | 18P8A | | 18P8B | |
| Speed Grade | Standard Speed | | | | High Speed | | | | Ultra High Speed | |
| Max. Access Time (ns) | STD | APL | STD | APL | STD | APL | STD | APL | STD | APL |
| | 35 | 40 | 35 | 40 | 25 | 30 | 25 | 30 | 15 | 20 |
| Max. Operating Current (mA) | 55 | | 90 | | | | 180 | | | |

STD = AMD "Standard" products
APL = AMD "Approved Products List" products

05799E/0
JANUARY 1988

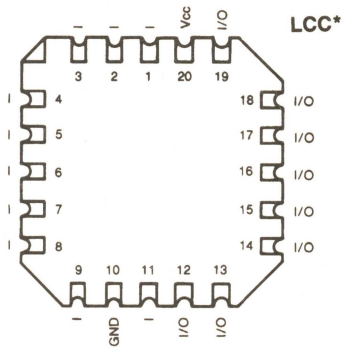
## Connection Diagrams

Top View




CD009210

Note: Pin 1 is marked for orientation.




CD009220

*Same Pinouts apply for PLCC.

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed/Power Option**
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**

AMPAL18P8 B P C –

**E. OPTIONAL PROCESSING**
Blank = Standard processing

**D. TEMPERATURE RANGE**
C = Commercial (0 to +70°C)
E = Extended Commercial (–55 to +125°C)

**C. PACKAGE TYPE**
P = 20-Pin Plastic DIP (PD 020)
D = 20-Pin Ceramic DIP (CD 020)
J = 20-Pin Plastic Leaded Chip Carrier (PL 020)
L = 20-Pin Ceramic Leadless Chip Carrier (CL 020)

**B. SPEED/POWER OPTION**
B = Ultra High Speed/Standard Power
A = High Speed/Standard Power
AL = High Speed/Half Power
L = Standard Speed/Half Power
Q = Standard Speed/Quarter Power

**A. DEVICE NUMBER/DESCRIPTION**
AmPAL18P8
20-Pin IMOX Programmable Array Logic

| Valid Combinations | |
|---|---|
| AMPAL18P8B | PC, DC, DE, JC, LC, LE |
| AMPAL18P8A | |
| AMPAL18P8AL | |
| AMPAL18P8L | |
| AMPAL18P8Q | |

#### Valid Combinations

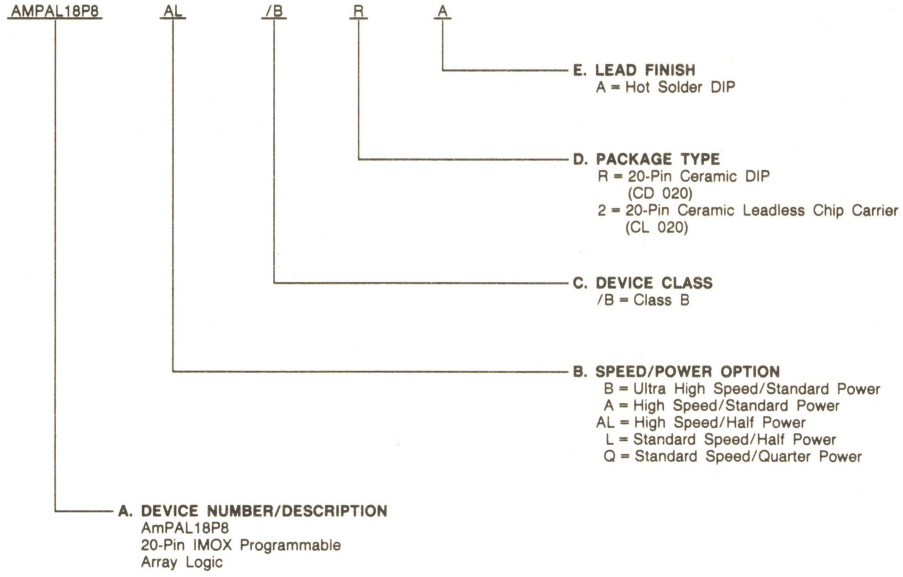
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information (Cont'd.)

### APL Products

AMD products for Aereospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-833C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

**A. Device Number**
**B. Speed/Power Option**
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL18P8B | /BRA, /B2A |
| AMPAL18P8A | |
| AMPAL18P8AL | |
| AMPAL18P8L | |
| AMPAL18P8Q | |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

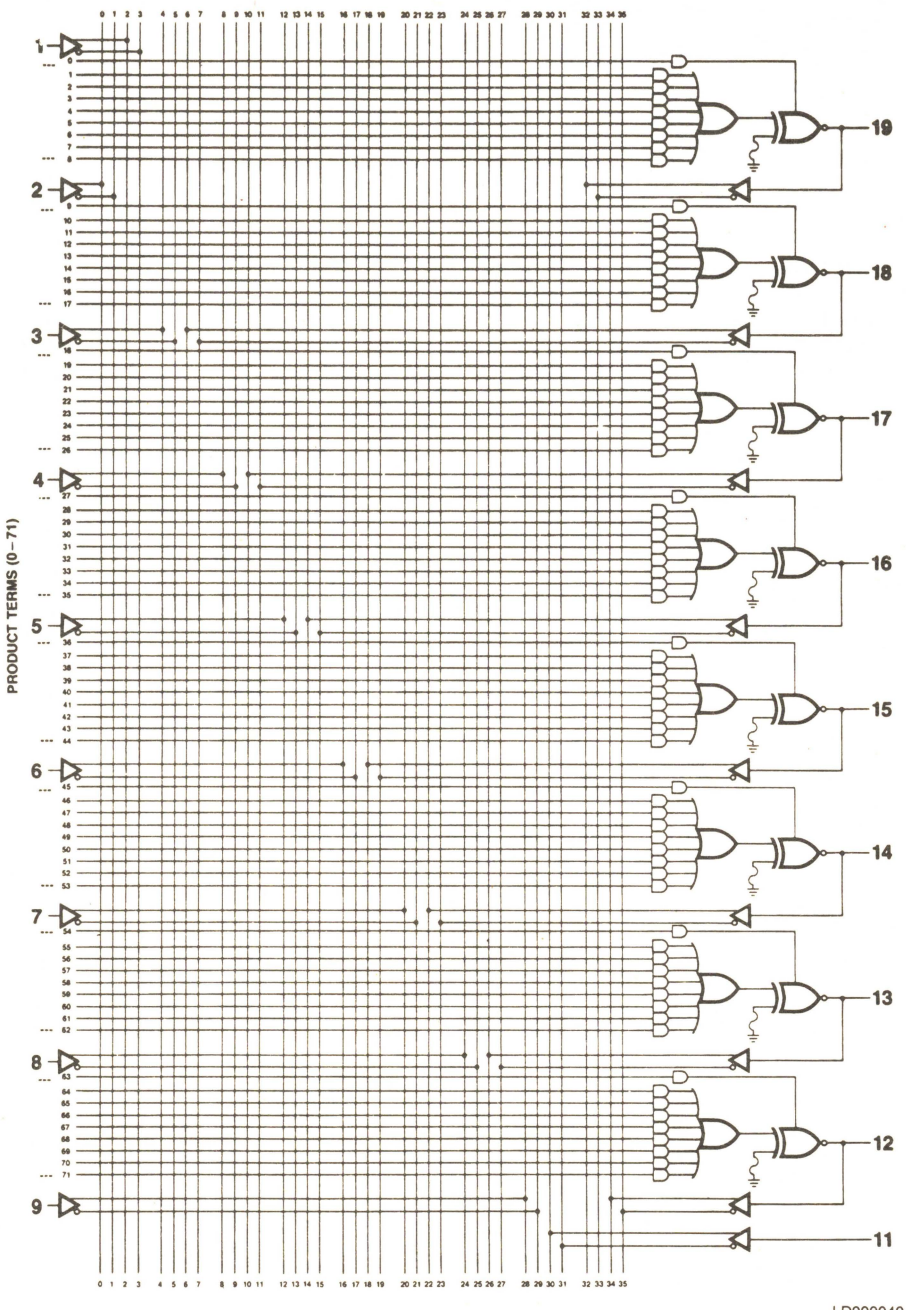
**Group A Tests**

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11

The AmPAL18P8 can be used as a functional and pin-for-pin replacement for each of the following 20-pin devices:

PAL10H8
PAL12H6
PAL14H4
PAL16H2
PAL10L8
PAL12L6
PAL14L4
PAL16L2
PAL16P8

## Logic Diagram




LD000040

**Eighteen Array Inputs**
- 10 dedicated
- 8 bidirectional I/O

**Eight 8-Wide AND-OR Structures**
- Combinatorial outputs
- Programmable output enable for each output
- Programmable polarity on each output

## Functional Description

The AmPAL18P8 is a functionally enhanced Programmable Array Logic (PAL) device. The Block Diagram on page ?? shows the basic architecture of the AmPAL18P8. There are up to eighteen inputs and eight outputs available. The inputs are connected to a programmable AND array which contains 72 logical product terms. Initially the AND gates are connected, via fuses, to both the true and complement of every input. By selective programming of fuses, the AND gates may be "connected" to only the true input (by blowing the complement fuse) to only the complement input (by blowing the true fuse), or to neither type of input (by blowing both fuses), establishing a logical "don't care." When both the true and complement fuses are left intact, a logical false results on the output of the AND gate. An AND gate with all fuses blown will assume the logical true state.

The AmPAL18P8 has a possible maximum of 18 input pins, two more than previous 20-pin PAL devices. The extra inputs extend the functional capabilities of the device, which reduces design limitations, making it easier to design with and more flexible.

The AmPAL18P8 can be programmed with more complex logic equations due to the eight product terms and one control term for each output. The control terms also allow for each of the eight bi-directional I/Os to be three-stated, greatly expanding the realm of design possibilities.

The eight bi-directional I/O pins enhance the usefulness of the AmPAL18P8 by allowing for greater complexity of logic equations and hence more logic power.

The AmPAL18P8 also has programmable output polarity, giving the designer the choice of either active HIGH or active LOW on each of the eight outputs. This simplifies the task of programming the AmPAL18P8 and allows more freedom in optimizing the logic functions. The high-speed version of the AmPAL18P8 boasts 15 ns maximum input-to-output propagation delay, and creates new possibilities for the use of programmable logic devices in a wide variety of applications.

The AmPAL18P8 is manufactured using Advanced Micro Devices' IMOX oxide isolation process. This advanced process permits an increase in density and a decrease in internal capacitance, resulting in the fastest possible programmable logic devices. The AmPAL18P8 is fabricated with AMD's fast-programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern.

Platinum-Silicide was selected as the fuse-link material to achieve a well-controlled melt rate, resulting in large non-conductive gaps that ensure very stable, long-term reliability. Extensive operating testing has proven that this low-field, large gap technology offers high reliability.

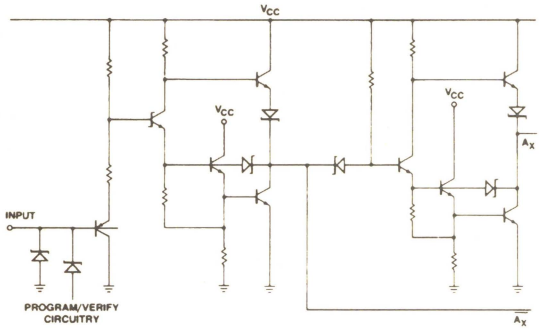
The AmPAL18P8 has been designed with extensive internal test circuitry that allows the programming and operating circuitry in the part to be thoroughly tested at the factory before programming. This assures excellent programming yield and functional performance to data sheet parameters after programming. The Post-Programming Functional Yield (PPFY) for this device is consistently better than 99.9%.

## Programmer/Development Systems

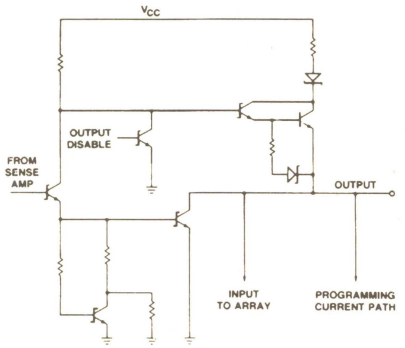
Refer to Programmer Reference Guide

## Input/Output Diagrams

### Input




### Output




IC000803

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage
with Respect to Ground ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs
(except during programming)..... −0.5 to +$V_{CC}$ Max.
DC Voltage Applied to
Outputs During Programming ..... 16 V
Output Current Into Outputs
During Programming
(Maximum duration of 1 second) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5.0 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) ..... 0 to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | 18P8A, 18P8B | COM'L | 2.4 | 3.5 | | Volts |
| | | | | 18P8L, 18P8AL | | | | | |
| | | | $I_{OH}$ = −2 mA | 18P8Q | | | | | |
| | | | $I_{OH}$ = −2 mA | (all versions) | MIL | | | | |
| $V_{OL}$ | Output LOW Voltage | | $I_{OL}$ = 24 mA | 18P8A, 18P8B | COM'L | | | 0.50 | Volts |
| | | | | 18P8L, 18P8AL | | | | | |
| | | | $I_{OL}$ = 12 mA | 18P8Q | | | | | |
| | | | $I_{OL}$ = 12 mA | A, B, AL, L | MIL | | | | |
| | | | $I_{OL}$ = 8 mA | 18P8Q | | | | | |
| $V_{IH}$ (Note 2) | Input HIGH level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | | 2.0 | | | Volts |
| $V_{IL}$ (Note 2) | Input LOW level | Guaranteed Input Logical LOW Voltage for All inputs | | | | | | 0.8 | Volts |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | | −20 | −100 | µA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | | 25 | µA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | | −30 | −60 | −90 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | | 18P8A, 18P8B | | | | 180 | mA |
| | | | | 18P8L, 18P8AL | | | | 90 | |
| | | | | 18P8Q | | | | 55 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | | −0.9 | −1.2 | Volts |
| $I_{OZH}$ | Output Leakage Current | Vcc = Max., $V_{IN}$ = $V_{IL}$ or $V_{IH}$ | $V_O$ = 2.7 V | | | | | 100 | µA |
| $I_{OZL}$ | | | $V_O$ = 0.4 V | | | | | −250 | |

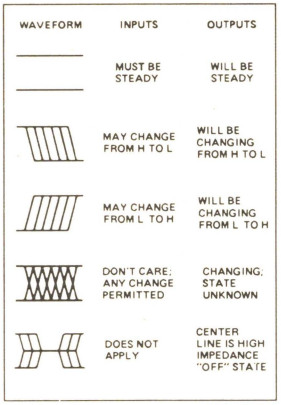
Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.

### Capacitance

| Parameter Symbol | Parameter Description | Test Conditions | Typ. | Units |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz | 6 | pF |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | 9 | |

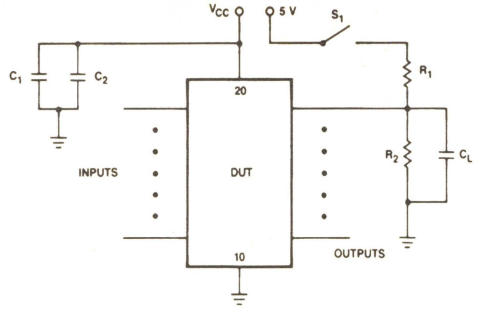
Note: These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

## Key to Switching Waveforms




KS000010

## Switching Test Circuit




TC003050

Note: $C_1$ and $C_2$ are to bypass $V_{CC}$ to ground during testing.

**TEST OUTPUT LOADS**

| Power Grade | $R_1$ (Ω) | | $R_2$ (Ω) | | $C_L$ (pF) | $C_1$ (μF) | $C_2$ (μF) |
|---|---|---|---|---|---|---|---|
| | STD | APL | STD | APL | STD/APL | STD/APL | STD/APL |
| 18P8B A AL L | 200 | 390 | 390 | 750 | 50 | 0.1 | 0.01 |
| 18P8Q | 390 | 600 | 750 | 1200 | 50 | 0.1 | 0.01 |

STD = AMD "Standard" products
APL = AMD "Approved Products List" products

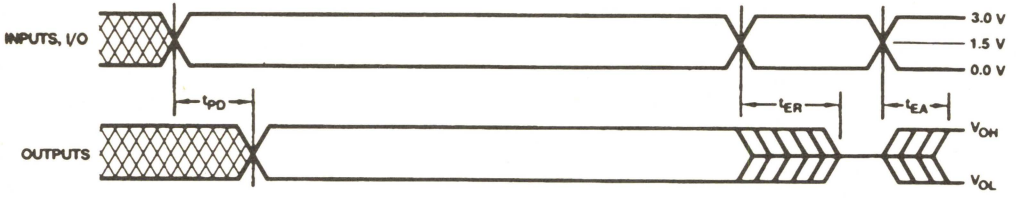
## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

| Parameter | Description | Commercial | | | | | | Military/Extended | | | | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 18P8B | | 18P8A/AL | | 18P8L/Q | | 18P8B | | 18P8A/AL | | 18P8L/Q | | |
| | | Typ. | Max. | Typ. | Max. | Typ. | Max. | Typ. | Max. | Typ. | Max. | Typ. | Max. | |
| $t_{PD}$ | Input to Output Delay | 12 | 15 | 15 | 25 | 25 | 35 | 12 | 20 | 15 | 30 | 25 | 40 | ns |
| $t_{EA}$ | Input to Output Enable | 12 | 15 | 15 | 25 | 25 | 35 | 12 | 20 | 15 | 30 | 25 | 40 | ns |
| $t_{ER}$ | Input to Output Disable | 12 | 15 | 15 | 25 | 25 | 35 | 12 | 20 | 15 | 30 | 25 | 40 | ns |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L$ = 50 pF.
3. For three-state output, output enable times are tested with $C_L$ = 50 pF to the 1.5 V level; $S_1$ is open for high-impedance to HIGH tests and closed for high-impedance to LOW tests. Output disable times are tested with $C_L$ = 5 pF. HIGH to high-impedance tests are made to an output voltage of $V_{OH}$ – 0.5 V with $S_1$ open; LOW to high-impedance tests are made to the $V_{OL}$ + 0.5 V level with $S_1$ closed.

## Switching Waveform




WF021820

# AmPALC29MA16

24-Pin $E^2$-Based CMOS Programmable Array Logic

ADVANCE INFORMATION

## Distinctive Characteristics

- High-performance semi-custom logic replacement; Electrically Erasable ($E^2$) technology allows reprogrammability
- 16 bidirectional user-programmable I/O logic macrocells for Combinatorial/Registered/Latched operation
- Output Enable controlled by a pin or product terms
- Variable product term distribution for increased design flexibility
- Programmable clock selection with common pin clock/latch enable (LE) or individual product term clock/LE with LOW/HIGH clock/LE polarity
- Register/Latch PRELOAD permits full logical verification
- Available in high-speed ($t_{PD}$ = 35 ns, $f_{MAX}$ = 20 MHz) and standard-speed ($t_{PD}$ = 45 ns, $f_{MAX}$ = 15.0 MHz) versions
- 100% post-programming functional yield (PPFY), fast programming and excellent reliability assured through proven $E^2$PROM technology
- Full-function AC and DC testing at the factory
- 24-pin 300-mil DIP and 28-pin chip carrier packages

## General Description

The AmPALC29MA16 is a high-speed, $E^2$-based CMOS Programmable Array Logic device designed for general logic replacement in TTL or CMOS digital systems. It offers high-speed, low-power consumption, high programming yield, fast programming and excellent reliability. Programmable logic devices (PLDs) combine the flexibility of custom logic with the off-the-shelf availability of standard products, providing major advantages over other semicustom solutions such as gate arrays and standard cells, including reduced development time and low up-front development cost.

The AmPALC29MA16 uses the familiar sum-of-products (AND-OR) structure, allowing users to customize logic functions by programming the device for specific applications. It provides up to twenty-nine array inputs and sixteen outputs. It incorporates AMD's unique input/output logic macrocell which provides flexible input/output structure and polarity, flexible feedback selection, multiple Output Enable choices, and a programmable clocking scheme. The macrocells can be individually programmed as "Combinatorial", "Registered", or "Latched" with active-HIGH or active-LOW polarity. The flexibility of the logic macrocells permits the system designer to tailor the device to particular application requirements.

Increased logic power has been built into the AmPALC29MA16 by providing a variable number of logical product terms per output. Eight outputs have four product terms each, four outputs have eight product terms each, and the other four outputs have twelve product terms each. This variable product-term distribution allows complex functions to be implemented in a single PAL device. Each output can be dynamically controlled by a common Output Enable pin or an individual Output Enable product terms. Each output can also be permanently enabled or disabled.

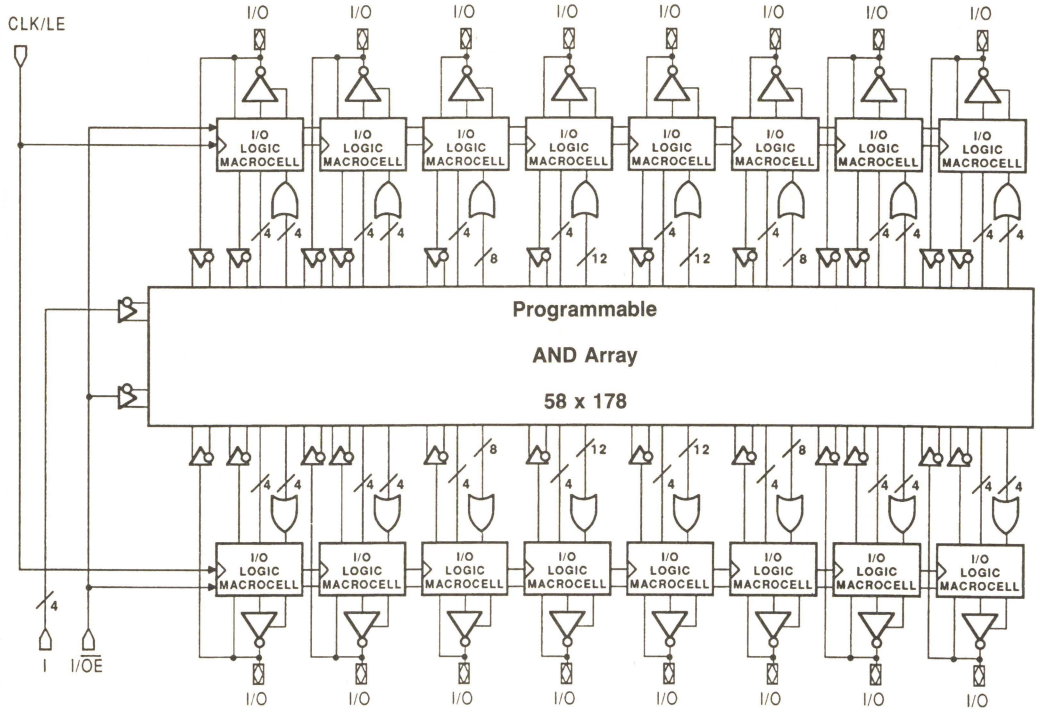
System operation has been enhanced by the addition of common asynchronous-PRESET and RESET product terms and a power-up RESET feature. The AmPALC29MA16 also incorporates PRELOAD and Observability functions which permit full logical verification of the design.

The AmPALC29MA16 is offered in the space-saving 300-Mil DIP package as well as chip carrier surface-mount packages.



08811B/0
JANUARY 1988

## Block Diagram

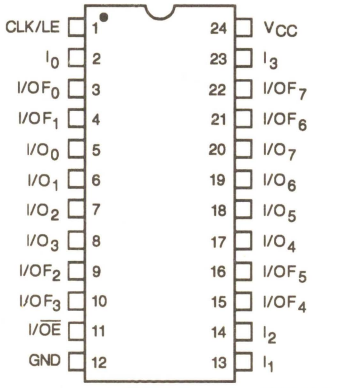



BD006860

## Connection Diagrams

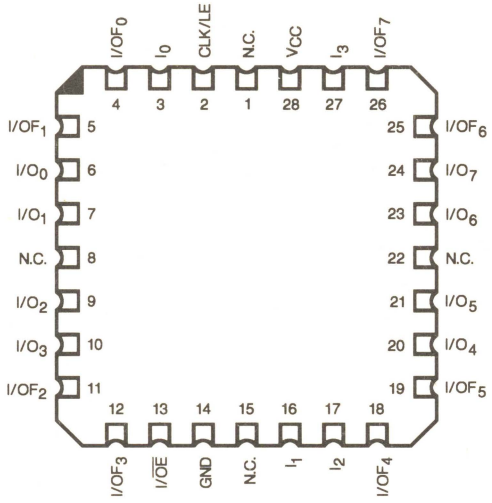
Top View

DIPs




CD010272

LCC*




CD010281

*Also available in PLCC. Pinouts identical to LCC.

Note: Pin 1 is marked for orientation.

5

## Pin Description

The following describes the functionality of all the pins on the 24-pin DIP. The 28-pin chip carrier has the same functionality with NO CONNECTS on pins 1,8,15,22.

**CLK/LE (PIN 1):**
Used as dedicated clock/latch enable pin for all registers/latches on the device if so selected. (See I/O Logic Macrocell Configurations.) This pin is a clock pin for macrocells configured as registers and a latch enable pin for macrocells configured as latches.

**I/$\overline{OE}$ PIN (PIN 11):**
Used as a dedicated input pin to the AND array or as the Output Enable control pin (Active LOW) for all macrocells with pin-controlled Output Enable selected.

**$I_0$-$I_3$ (PINS 2,13,14,23):**
Dedicated input pins.

**$I/OF_0$-$I/OF_7$ (PINS 3,4,9,10,15,16,21,22):**
Eight bidirectional I/O pins with two independent feedback paths to the AND array. The first feedback path is a dedicated I/O pin feedback to the AND array for combinatorial input. The second feedback path consists of direct register/latch feedback to the array (see Figure 1).

**$I/O_0$-$I/O_7$ (PINS 5,6,7,8,17,18,19,20):**
Eight bidirectional I/O pins with user-programmable register/latch or I/O pin feedback to the AND array (see Figure 1).

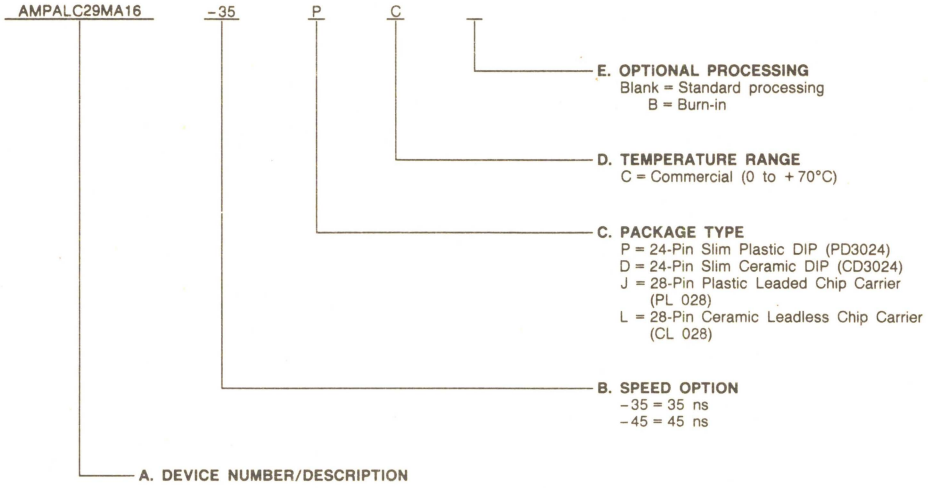
**$V_{CC}$ (PIN 24):**
Supply Voltage

**GND (PIN 12):**
Circuit Ground

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AmPALC29MA16-35, -45 | PC, DC, DCB, JC, LC, LCB |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

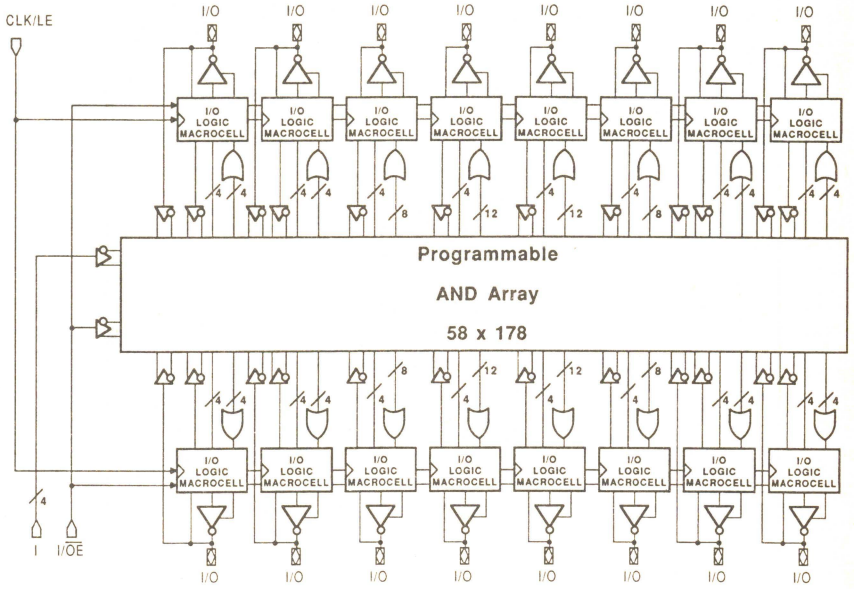
## Functional Description

### Inputs

The AmPALC29MA16 has 29 inputs to drive each product term (up to 58 inputs with both TRUE and complement versions available to the AND array) as shown in the block diagram below. Of these 29 inputs, 4 are dedicated inputs, 16 are from 8 I/O logic macrocells with 2 feedbacks, 8 are from other I/O logic macrocells with single feedback and 1 is for I/$\overline{OE}$ input.

Initially the AND-array gates are disconnected from all the inputs. This condition represents a logical TRUE to the AND array. By selectively programming the $E^2$cells, the AND array may be connected to either the TRUE input or the complement input. When both the TRUE and complement inputs are connected, a logical FALSE results at the output of the AND gate.

### Block Diagram




### Product Terms

The degree of programmability and complexity of a PAL device is determined by the number of connections that form the programmable-AND and OR gates. Each programmable-AND gate is called a product term. The AmPALC29MA16 has 178 product terms. 112 of these product terms provide logic capability and others are architectural product terms. Among the control product terms, 1 is for Observability, and 1 is for PRELOAD. The Output Enable of each macrocell can be programmed to be controlled by a common Output Enable pin or an individual product term. It may be also permanently enabled or permanently disabled. In addition, independent product terms for each marcocell control PRESET, RESET and CLK/LE.

Each product term on the AmPALC29MA16 consists of a 58-input AND gate. The outputs of these AND gates are connected to a fixed-OR plane. Product terms are allocated to OR gates in a variable distribution across the device ranging from 4-to-12 wide, with an average of 7 logical product terms per output. Increased number of product terms per output allows more complex functions to be implemented in a single PAL device. This flexibility aids in implementing functions such as counters, exclusive-OR functions, or complex state machines, where different states require different numbers of product terms.

Individual asynchronous-PRESET and RESET product terms are connected to all Registered/Latched inputs/outputs.

When the asynchronous-PRESET product term is asserted (HIGH) all the registers/latches will immediately be loaded with a HIGH, independent of the clock. When the asynchronous-RESET product term is asserted (HIGH) all the registers/latches will be immediately loaded with a LOW, independent of the clock. The actual output state will depend on the macrocell polarity selection. The latches must be in latched mode (not transparent mode) for the RESET/PRESET, PRELOAD, and power-up RESET modes to be meaningful.

### Input/Output Logic Macrocells

The I/O logic macrocell allows the user the flexibility of defining the architecture of each input or output on an individual basis. It also provides the capability of using the associated pin either as an input or an output.

The AmPALC29MA16 has 16 macrocells, one for each I/O pin. Each I/O macrocell can be programmed for combinatorial, registered or latched operation (see Figure 1). Combinatorial output is desired when the PAL device is used to replace combinatorial glue logic. Registers are used in synchronous logic applications while latches are used in asynchronous applications where speed is critical. The output polarity for each macrocell in each of the three modes of operation is user-selectable, allowing complete flexibility of the macrocell configuration.

5

Eight of the macrocells ($I/OF_0$-$I/OF_7$) have two independent feedback paths to the AND array (see Figure 1). The first is a dedicated I/O pin feedback to the AND array for combinatorial input. The second path consists of a direct register/latch feedback to the array. If the pin is used as a dedicated input using the first feedback path, the register/latch feedback path is still available to the AND array. This path provides the capability of using the register/latch as a buried state register/latch. The other eight macrocells have a single feedback path to the AND array. This feedback is user-selectable as either an I/O pin or a register/latch feedback.

Each macrocell can provide true input/output capability. The user can select each macrocell register/latch to be driven by either the output generated by the AND-OR array or the I/O pin. When the I/O pin is selected as the input, the feedback path provides the register/latch input to the array. When used as an input, each macrocell is also user-programmable for registered, latched, or combinatorial input.

The AmPALC29MA16 has one dedicated CLK/LE pin and an individual CLK/LE product term. All macrocells have a programmable select to choose between these two as the clock or the latch enable signal. These signals are clock signals for macrocells configured as registers and latch enable signal for macrocells configured as latches. The polarity of these CLK/LE signals is also individually programmable. Thus different registers can be driven by multiple clocks and clock phases.

The Output-Enable mode of each of the macrocells can be selected by the user. The I/O pin can be configured as an output pin (permanently enabled) or as an input pin (permanently disabled). It can also be configured as a dynamic I/O controlled by the Output Enable pin or by product term.

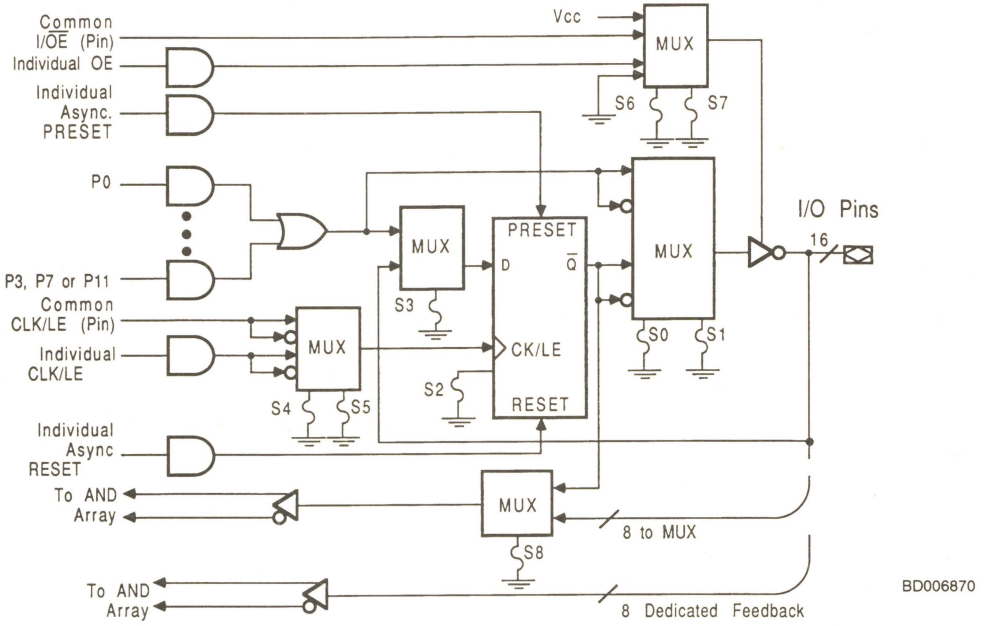



**Figure 1. AmPALC29MA16 I/O Macrocell**

### I/O Logic Macrocell Configuration

AMD's unique I/O macrocell offers major benefits through its versatile, programmable input/output cell structure, multiple clock choices, flexible Output Enable and feedback selection. Eight I/O macrocells with single feedback contain nine $E^2$cells, while the other eight macrocells contain eight $E^2$cells for programming the input/output functions (see Table 1, Figure 2).

$E^2$cell S1 controls whether the macrocell will be combinatorial or registered/latched. S0 controls the output polarity (active-HIGH or active-LOW). S2 determines whether the input/output is a register or a latch. S3 allows the use of the macrocell as an input register/latch or as an output register/latch. It selects the direction of the data path through the register/latch. If connected to the usual AND-OR array output, the register/latch is an output connected to the I/O pin. If connected to the I/O pin, the register/latch becomes an input register/latch to the AND array using the feedback data path.

Programmable $E^2$cells S4 and S5 allow the user to select one of the four CLK/LE signals for each macrocell. S6 and S7 are used to control Output Enable as pin controlled, product term controlled, permanently enabled or permanently disabled. S8 is a feedback multiplexer for the macrocells with a single feedback path only.

In the virgin erased state (charged, disconnected), an architectural cell is said to have a value of "1" ; In the programmed state (discharged, connected), an architectural cell is said to have a value of "0".

## Table 1. AmPALC29MA16 I/O Logic Macrocell Architecture Selections

| S3 | I/O Cell |
|---|---|
| 1 | Output Cell |
| 0 | Input Cell |

| S2 | Storage Element |
|---|---|
| 1 | Register |
| 0 | Latch |

| S1 | Output Type |
|---|---|
| 1 | Combinatorial |
| 0 | Register/Latch |

| S0 | Output Polarity |
|---|---|
| 1 | Active LOW |
| 0 | Active HIGH |

| S8 | Feedback* |
|---|---|
| 1 | Register/Latch |
| 0 | I/O |

*Applies to macrocells with single feedback only.

TC003961

## Table 1. AmPALC29MA16 I/O Logic Macrocell Clock Polarity & Output Enable Selections
(Cont'd.)

| S4 | S5 | Clock Edge/Latch Enable Level |
|---|---|---|
| 1 | 1 | CLK/LE pin positive-going edge, active-HIGH LE |
| 1 | 0 | CLK/LE pin negative-going edge, active-LOW LE |
| 0 | 1 | CLK/LE PT positive-going edge, active-HIGH LE |
| 0 | 0 | CLK/LE PT negative-going edge, active-LOW LE |

| S6 | S7 | Output Buffer Control |
|---|---|---|
| 1 | 1 | Pin-Controlled 3-State Enable |
| 1 | 0 | PT-Controlled 3-State Enable |
| 0 | 1 | Permanently Enabled (Output only) |
| 0 | 0 | Permanently Disabled (Input only) |

TC003972

1 = Erased State (Charged or disconnected)
0 = Programmed State (Discharged or connected)

## Some Possible Configurations of the Input/Output Logic Macrocell

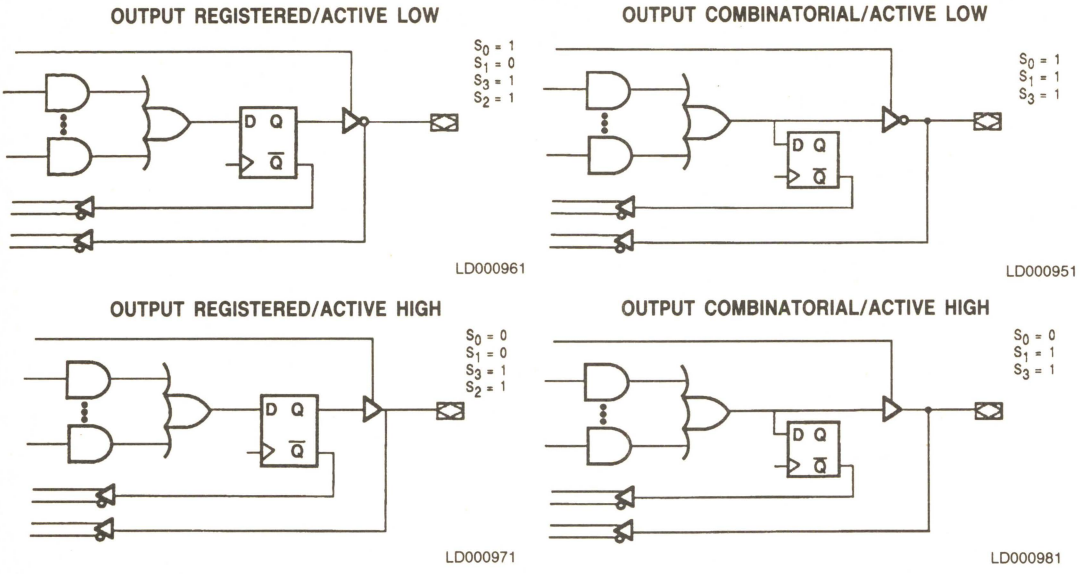



Figure 2a. Dual Feedback Macrocells

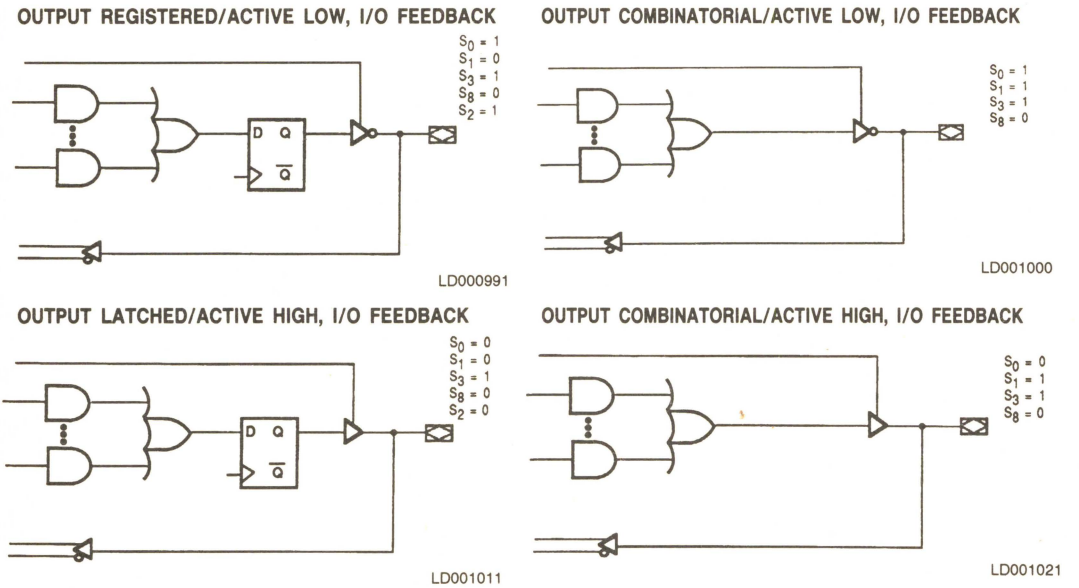



Figure 2b. Single Feedback Macrocells

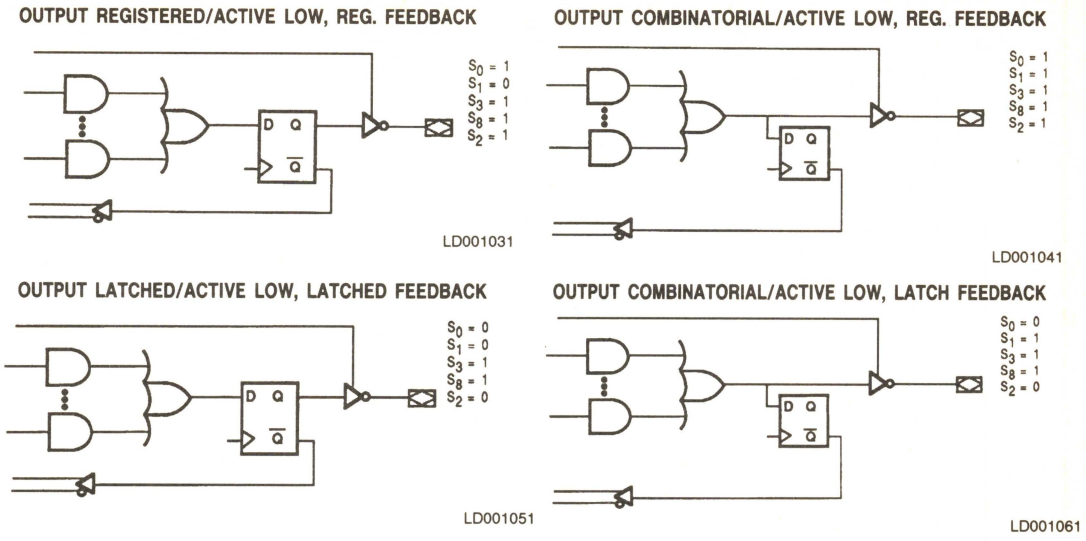



**Figure 2b. Single Feedback Macrocells** (Cont'd.)

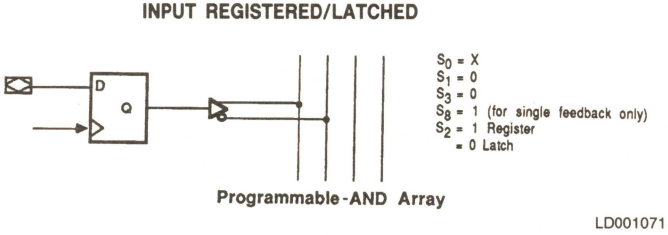
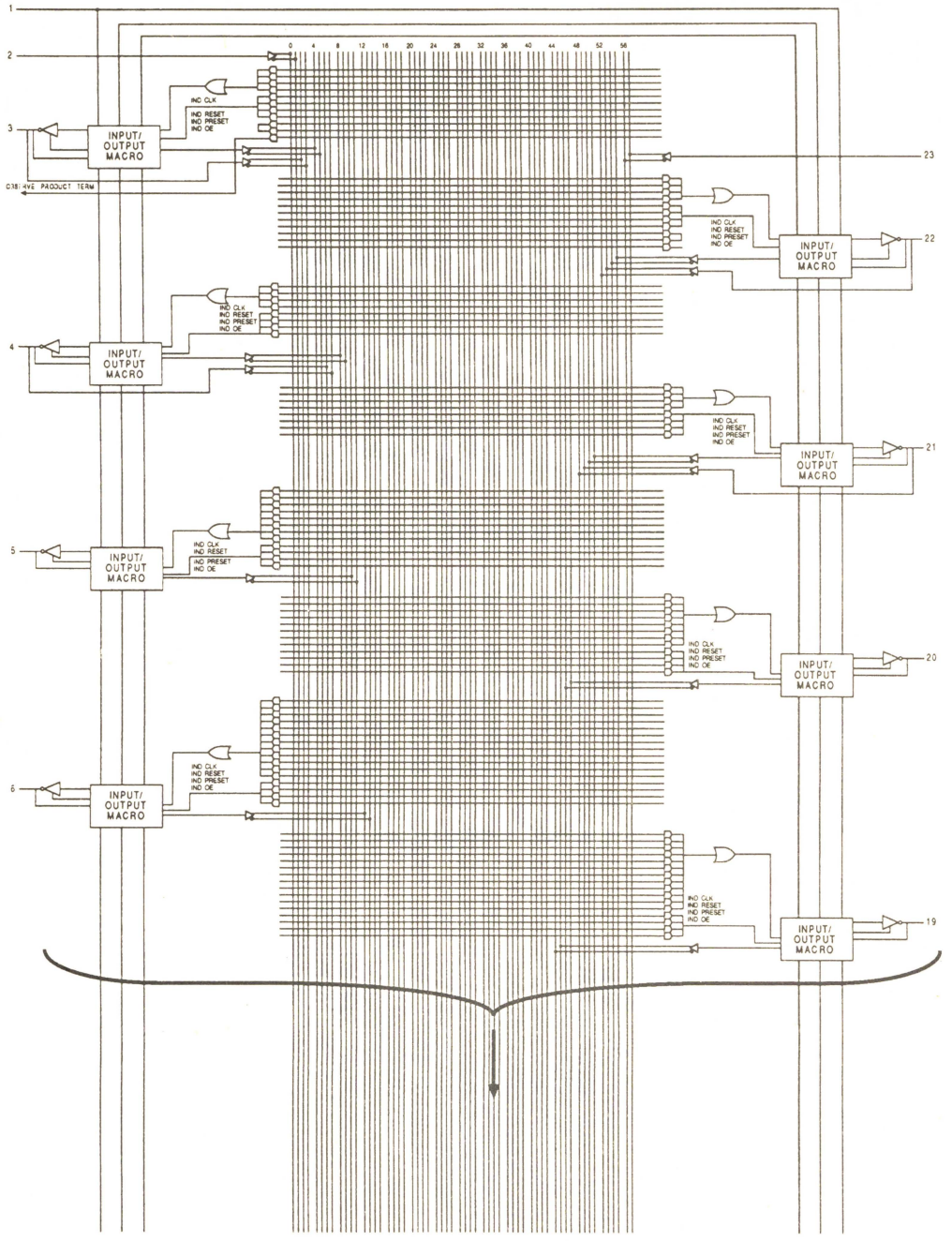



**Figure 2c. All Macrocells**

5




LD001320

Figure 3. AmPALC29MA16 Logic Diagram

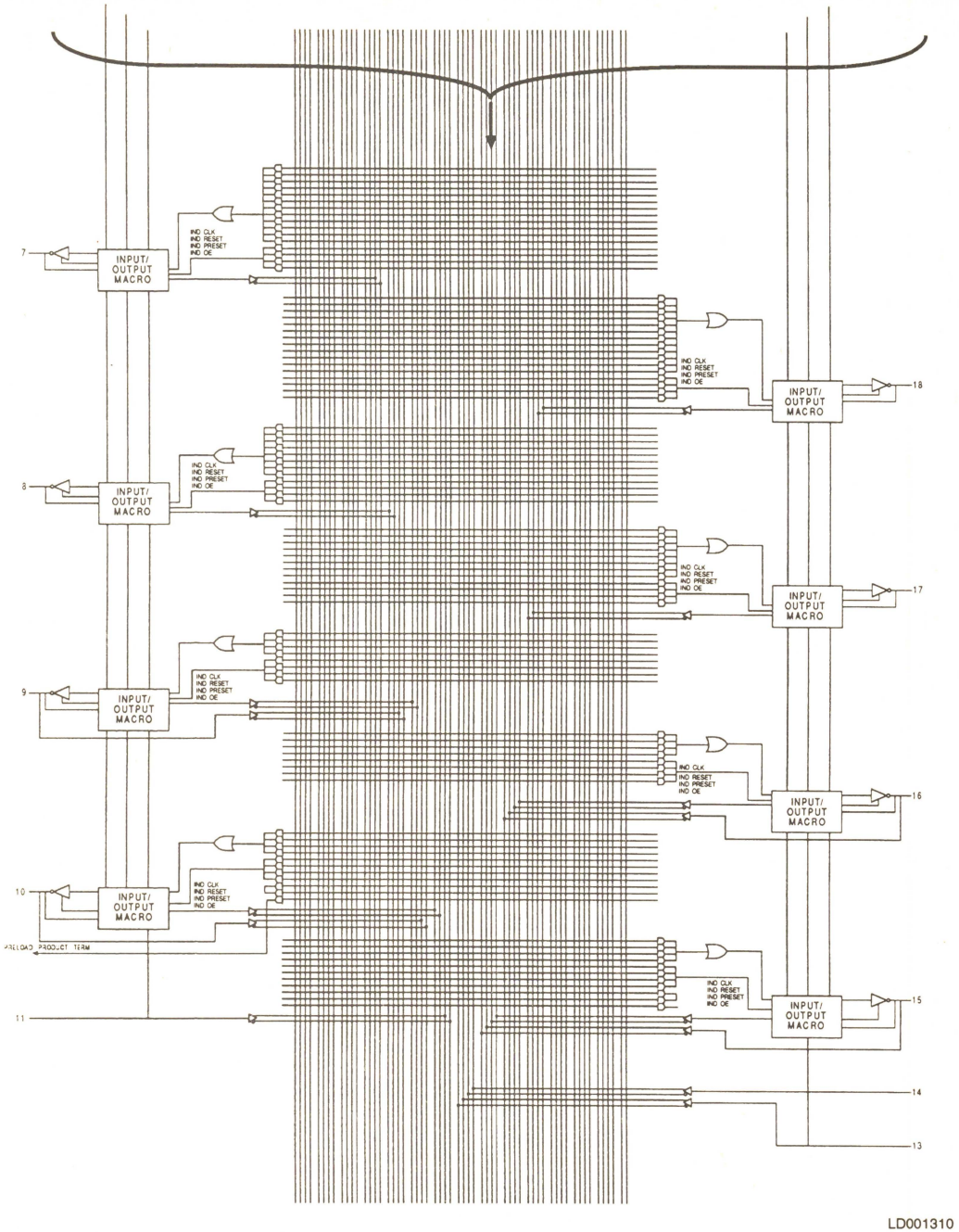



**Figure 3. AmPALC29MA16 Logic Diagram** (Cont'd.)

## Designed in Testability and Debugging

### PRELOAD

To simplify testing, the AmPALC29MA16 is designed with PRELOAD circuitry that provides an easy method for testing logical functionality. Both product-term controlled and supervoltage-enabled PRELOAD modes are available. This offers even more test capability than previously implemented in AMD's PAL devices. The TTL-level PRELOAD product term can be useful during debugging, where supervoltages may not be available.

PRELOAD allows any arbitrary state value to be loaded into the registers/latches of the device. A typical functional-test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the devices inputs into any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state, or "next state", which can be checked to validate the transition from the "present state". In this way any transition can be checked.

Since PRELOAD can provide the capability to go directly to any desired arbitrary state, test sequences may be greatly shortened. Also, all possible states can be tested, thus greatly reducing test time and development costs and guaranteeing proper in-system operation.

### Observability

The output register/latch observability product term, when asserted, suppresses the combinatorial output data from appearing on the I/O pin and allows the observation of the contents of the register/latch on the output pin for each of the logic macrocells. This unique feature allows for easy debugging and tracing of the buried state machines. In addition, a capability of supervoltage observability is also provided.

### Power-Up Reset

All the device registers/latches have been designed to reset during device power-up. Following the power-up, all registers/latches will be cleared (Q = 0), setting the outputs to a state determined by the output select multiplexer. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

### Security Cell

A security cell is provided on each device to prevent unauthorized copying of the user's proprietary logic design. Once programmed, the security cell disables the programming, verification, PRELOAD, and the observability modes. The only way to erase the protection cell is by charging the entire array and architecture cells, in which case no proprietary design can be copied. (This cell should be programmed only after the rest of the device has been completely programmed and verified.)

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Ambient Temperature under bias..... −55 to +125°C
Supply Voltage with
Respect to Ground..... −0.5 V to +7.0 V
DC Output Voltage..... −0.5 V to $V_{CC}$ + 0.5 V
DC Input Voltage
(Except Pin I/$\overline{OE}$) ..... −0.5 V to $V_{CC}$ + 0.5 V
DC Input Voltage (Pin I/$\overline{OE}$) ..... −0.6 V to +17 V
DC Input Current ..... −1 mA to +1 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$)..... 0°C to +70°C
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Consult Factory for Military Specifications

## DC Characteristics over operating range unless otherwise specified

### HCT Devices**

| Parameter Symbol | Parameter Description | Test Conditions | | Min. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$<br>$I_{OH}$ = −2 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 6 mA | | 0.5 | V |
| | | | $I_{OL}$ = 4 mA | | 0.33 | |
| | | | $I_{OL}$ = 20 μA | | 0.1 | |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Logic HIGH for all Inputs | | 2.0 | | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Logic LOW for all Inputs | | | 0.8 | V |
| $I_I$ | Input Leakage Current | $V_{IN}$ = 0 to 5.5 V, $V_{CC}$ = Max. | | | 10 | μA |
| $I_O$ | Output Leakage Current | $V_{IN}$ = 0 to 5.5 V, $V_{CC}$ = Max. | | | 10 | μA |
| $I_{CCOP}$ | Operating Current Supply | f = $f_{MAX}$, Outputs Open ($I_O$ = 0) | | | 120 | mA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_O$ = 0 V | | −30 | −90 | mA |

### Capacitance

| Parameter Symbol | Parameter Description | Test Conditions | Typ. | Units |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{CC}$ = 5.00 V., $T_A$ = 25°C<br>$V_{IN}$ = 0 V @ f = 1 MHz | 5 | pF |
| $C_{OUT}$ | Output Capacitance | | 8 | |

Note: These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

** Consult Factory for DC specifications for HC Devices.

5

## Switching Characteristics

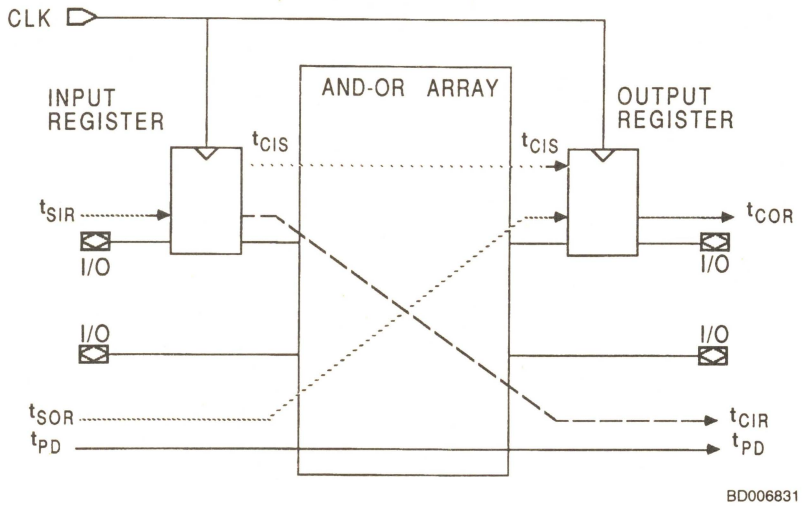
over operating range unless otherwise specified; all values are determined under the loading of one TTL gate and a capacitance of 50 pF

| Parameter Number | Parameter Symbol | Parameter Description | -35 | | -45 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| **REGISTERED OPERATION (Numbers 1 through 20)** | | | | | | | |
| 1 | $t_{PD}$ | Input or I/O Pin to Combinatorial Output | | 35 | | 45 | ns |
| **Output Register – Pin Clock** | | | | | | | |
| 2 | $t_{SOR}$ | Input or I/O Pin to Output Register Setup | 27 | | 34 | | ns |
| 3 | $t_{COR}$ | Output Register Clock to Output | | 23 | | 32 | ns |
| 4 | $t_{HOR}$ | Data Hold Time for Output Register | 0 | | 0 | | ns |
| **Output Register – Product Term Clock** | | | | | | | |
| 5 | $t_{SORP}$ | I/O Pin or Input to Output Register Setup | 20 | | 24 | | ns |
| 6 | $t_{CORP}$ | Output Register Clock to Output | | 45 | | 56 | ns |
| 7 | $t_{HORP}$ | Data Hold Time for Output Register | 15 | | 20 | | ns |
| **Input Register – Pin Clock** | | | | | | | |
| 8 | $t_{SIR}$ | I/O Pin to Input Register Setup | 6 | | 8 | | ns |
| 9 | $t_{CIR}$ | Register Feedback Clock to Combinatorial Output | | 45 | | 58 | ns |
| 10 | $t_{HIR}$ | Data Hold Time for Input Register | 3 | | 4 | | ns |
| **Clock and Frequency** | | | | | | | |
| 11 | $t_{CIS}$ | Register Feedback (Pin Driven Clock) to Output Register/Latch (Pin Driven) Setup | 35 | | 45 | | ns |
| 12 | $t_{CISPP}$ | Register Feedback (PT Driven Clock) to Output Register/Latch (PT Driven) Setup | 45 | | 60 | | ns |
| 13 | $f_{MAX}$ | Maximum Frequency (Pin Driven) $1/(t_{SOR} + t_{COR})$ | | 20 | | 15 | MHz |
| 14 | $f_{MAXI}$ | Maximum Internal Frequency (Pin Driven) $1/t_{CIS}$ | | 28.5 | | 22.5 | MHz |
| 15 | $f_{MAXP}$ | Maximum Frequency (PT Driven) $1/(t_{SORP} + t_{CORP})$ | | 15.5 | | 12.5 | MHz |
| 16 | $f_{MAXIPP}$ | Maximum Internal Frequency (PT Driven) $1/t_{CISPP}$ | | 22.5 | | 16.5 | MHz |
| 17 | $t_{CWH}$ | Pin Clock Width HIGH | 12 | | 15 | | ns |
| 18 | $t_{CWL}$ | Pin Clock Width LOW | 12 | | 15 | | ns |
| 19 | $t_{CWHP}$ | PT Clock Width HIGH | 15 | | 20 | | ns |
| 20 | $t_{CWLP}$ | PT Clock Width LOW | 15 | | 20 | | ns |

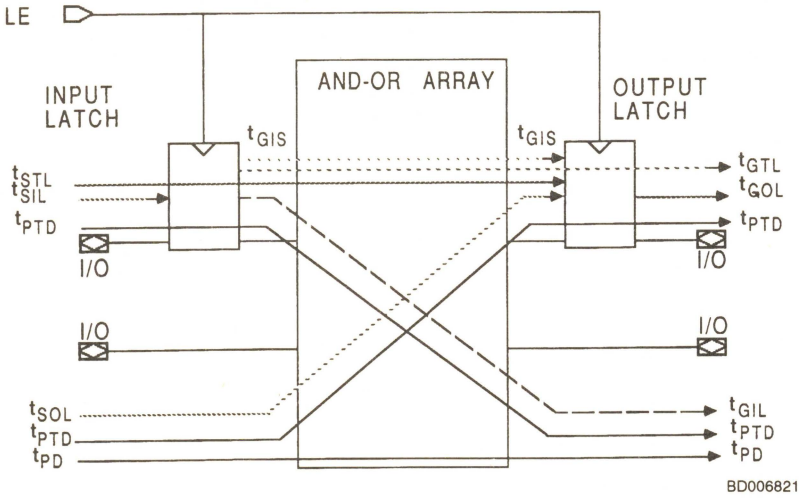
| Parameter Number | Parameter Symbol | Parameter Description | -35 | | -45 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| **LATCHED OPERATION (Numbers 21 through 39)** | | | | | | | |
| 21 | $t_{PD}$ | Input or I/O Pin to Combinatorial Output | | 35 | | 45 | ns |
| 22 | $t_{PTD}$ | Input or I/O Pin to Output via Transparent Latch | | 45 | | 55 | ns |
| **Output Latch – Pin LE** | | | | | | | |
| 23 | $t_{SOL}$ | Input or I/O Pin to Output Latch Setup | 27 | | 34 | | ns |
| 24 | $t_{GOL}$ | Latch Enable to Transparent Mode Output | | 23 | | 32 | ns |
| 25 | $t_{HOL}$ | Data Hold Time for Output Latch | 0 | | 0 | | ns |
| 26 | $t_{STL}$ | Input or I/O Pin to Output Latch Setup via Transparent Input Latch | 35 | | 45 | | ns |
| **Output Latch – PT LE** | | | | | | | |
| 27 | $t_{SOLP}$ | Input or I/O Pin to Output Latch Setup | 20 | | 24 | | ns |
| 28 | $t_{GOLP}$ | Latch Enable to Transparent Mode Output | | 45 | | 56 | ns |
| 29 | $t_{HOLP}$ | Data Hold Time for Output Latch | 15 | | 20 | | ns |
| 30 | $t_{STLP}$ | Input or I/O Pin to Output Latch Setup via Transparent Input Latch | 30 | | 35 | | ns |
| **Input Latch – Pin LE** | | | | | | | |
| 31 | $t_{SIL}$ | I/O Pin to Input Latch Setup | 6 | | 8 | | ns |
| 32 | $t_{GIL}$ | Latch Feedback, Latch Enable Transparent Mode to Combinatorial Output | | 45 | | 58 | ns |
| 33 | $t_{HIL}$ | Data Hold Time for Input Latch | 3 | | 4 | | ns |
| **Latch Enable** | | | | | | | |
| 34 | $t_{GIS}$ | Latch Feedback (Pin Driven) to Output Register/Latch (Pin Driven) Setup | 35 | | 45 | | ns |
| 35 | $t_{GISPP}$ | Latch Feedback (PT Driven) to Output Register/Latch (PT Driven) Setup | 45 | | 60 | | ns |
| 36 | $t_{GWH}$ | Pin Enable Width HIGH | 12 | | 15 | | ns |
| 37 | $t_{GWL}$ | Pin Enable Width LOW | 12 | | 15 | | ns |
| 38 | $t_{GWHP}$ | PT Enable Width HIGH | 15 | | 20 | | ns |
| 39 | $t_{GWLP}$ | PT Enable Width LOW | 15 | | 20 | | ns |

| Parameter Number | Parameter Symbol | Parameter Description | -35 | | -45 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| **RESET/PRESET & OUTPUT ENABLE (Numbers 40 through 49)** | | | | | | | |
| 40 | $t_{APO}$ | Input or I/O Pin to Output Register/Latch RESET/ PRESET | | 40 | | 55 | ns |
| 41 | $t_{AW}$ | Async. RESET/PRESET Pulse Width | 35 | | 45 | | ns |
| 42 | $t_{ARO}$ | Async. RESET/PRESET to Output Register/Latch Recovery | 30 | | 40 | | ns |
| 43 | $t_{ARI}$ | Async. RESET/PRESET to Input Register/Latch Recovery | 20 | | 30 | | |
| 44 | $t_{ARPO}$ | Async. RESET/PRESET to Output Register/Latch Recovery PT Clock/LE | 20 | | 25 | | ns |
| 45 | $t_{ARPI}$ | Async. RESET/PRESET to Input Register/Latch Recovery PT Clock/LE | 15 | | 20 | | ns |
| 46 | $t_{PZX}$ | I/$\overline{OE}$ Pin to Output Enable | | 30 | | 40 | ns |
| 47 | $t_{PXZ}$ * | I/$\overline{OE}$ Pin to Output Disable | | 30 | | 40 | ns |
| 48 | $t_{EA}$ | Input or I/O to Output Enable via PT | | 35 | | 45 | ns |
| 49 | $t_{ER}$ * | Input or I/O to Output Disable via PT | | 35 | | 45 | ns |

* Output disable times do not include test load RC time constants.

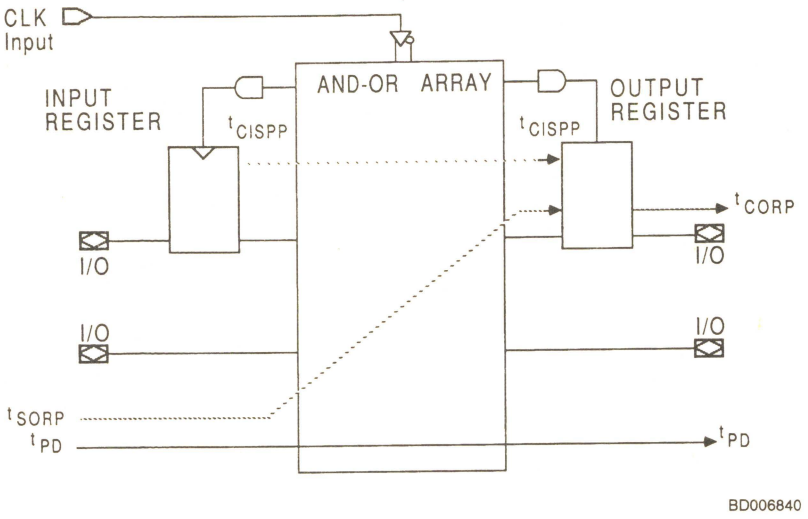



**Input/Output Register Specs (Pin CLK Reference)**

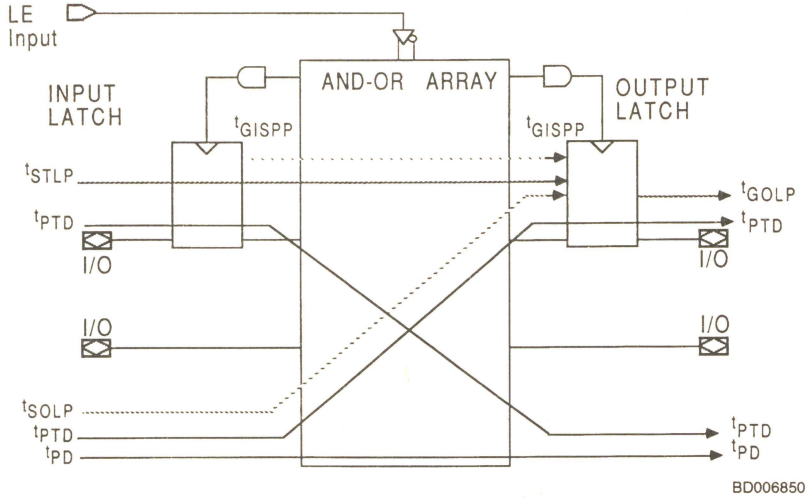



**Input/Output Latch Specs (Pin LE Reference)**

5

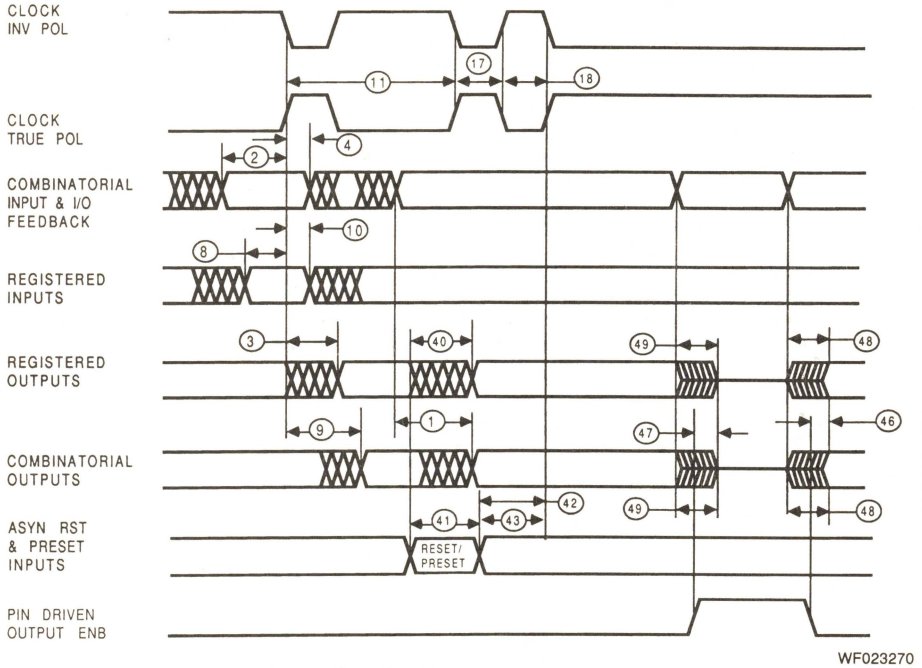



**Input/Output Register Specs (PT CLK Reference)**

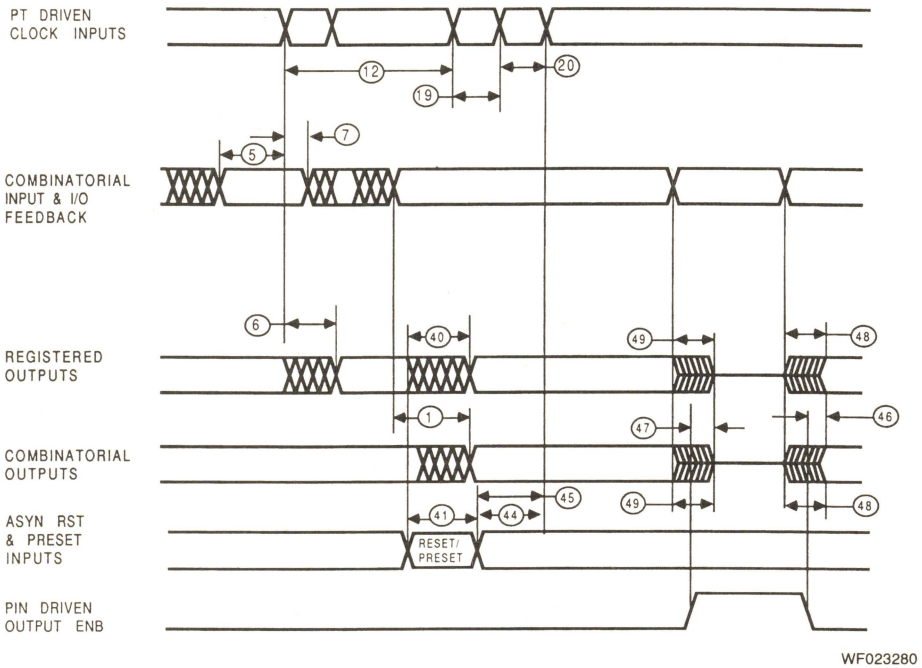



**Input/Output Latch Specs (PT LE Reference)**

## Switching Waveforms



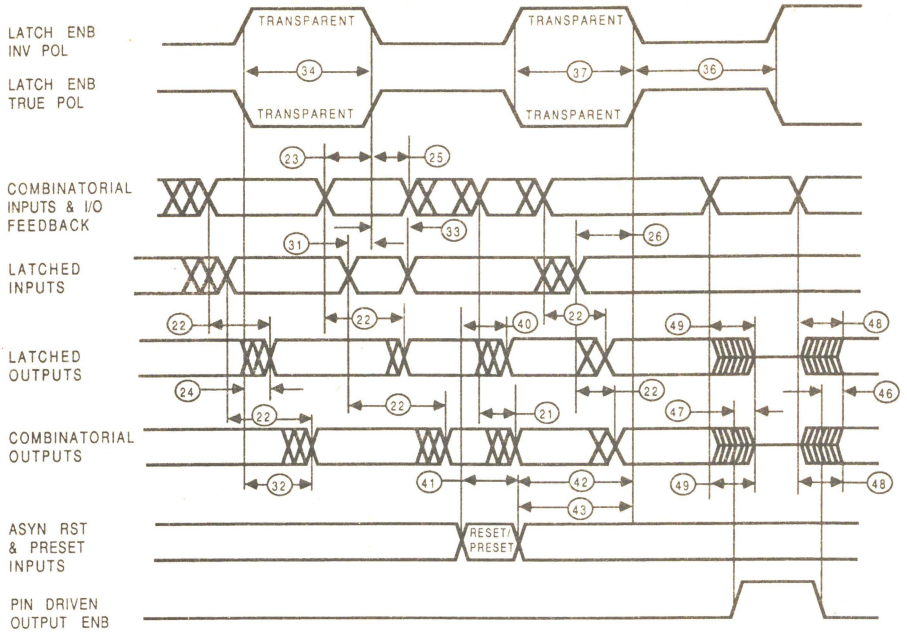

**Register (Pin CLK Reference)**




**Register (PT CLK Reference)**

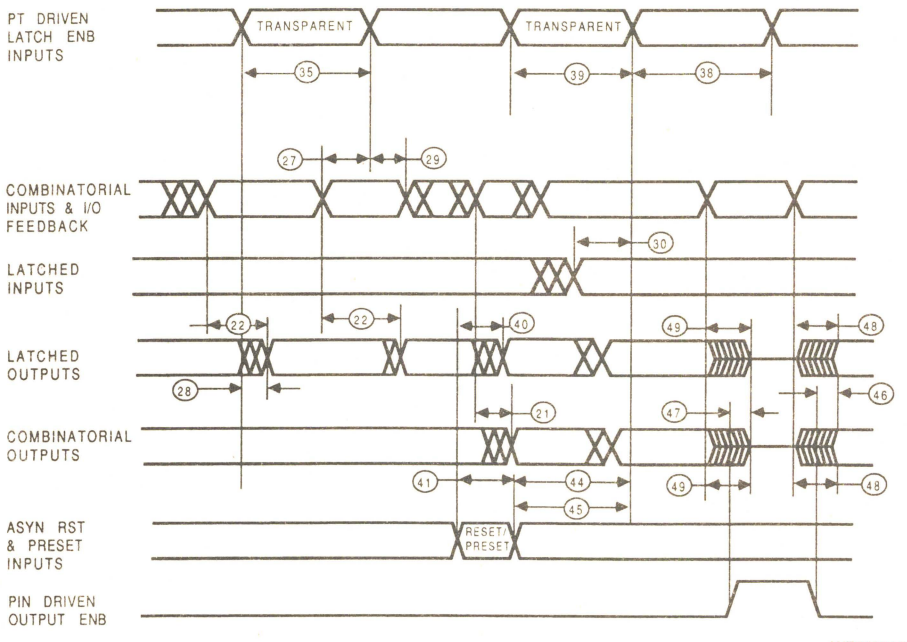
5

## Switching Waveforms (Cont'd.)




WF023290

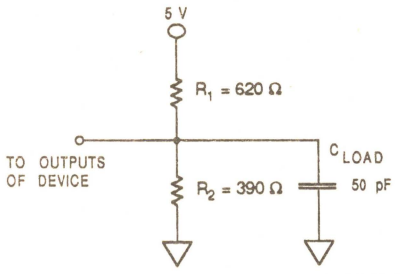
**Latch (Pin LE Reference)**




WF023300

**Latch (PT LE Reference)**

## Switching Test Circuit



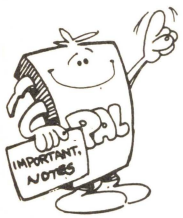

TC003951

## Key to Switching Waveforms

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| ——— | MUST BE STEADY | WILL BE STEADY |
| ▥ | MAY CHANGE FROM H TO L | WILL BE CHANGING FROM H TO L |
| ▨ | MAY CHANGE FROM L TO H | WILL BE CHANGING FROM L TO H |
| ▩ | DON'T CARE; ANY CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| ⋈ | DOES NOT APPLY | CENTER LINE IS HIGH IMPEDANCE "OFF" STATE |

KS000010

5


IMPORTANT
NOTES

# AmPALC29M16

24-Pin $E^2$-Based CMOS Programmable Array Logic

ADVANCE INFORMATION

## Distinctive Characteristics

- High-performance semi-custom logic replacement; Electrically Erasable ($E^2$) technology allows reprogrammability
- 16 bidirectional user-programmable I/O logic macrocells for Combinatorial/Registered/Latched operation
- Output Enable controlled by a pin or product terms
- Variable product term distribution for increased design flexibility
- Programmable clock selection with two clocks/latch enables (LEs) and LOW/HIGH clock/LE polarity
- Register/Latch PRELOAD permits full logical verification
- Available in high-speed ($t_{PD}$ = 35 ns, $f_{MAX}$ = 20 MHz) and standard-speed ($t_{PD}$ = 45 ns, $f_{MAX}$ = 15.0 MHz) versions
- 100% post-programming functional yield (PPFY), fast programming and excellent reliability assured through proven $E^2$PROM technology
- Full-function AC and DC testing at the factory
- 24-pin 300-mil DIP and 28-pin chip carrier packages

## General Description

The AmPALC29M16 is a high-speed $E^2$-based CMOS Programmable Array Logic device designed for general logic replacement in TTL or CMOS digital systems. It offers high-speed, low power consumption, high programming yield, fast programming and excellent reliability. Programmable logic devices (PLDs) combine the flexibility of custom logic with the off-the-shelf availability of standard products, providing major advantages over other semicustom solutions such as gate arrays and standard cells, including reduced development time and low up-front development cost.

The AmPALC29M16 uses the familiar sum-of-products (AND-OR) structure, allowing users to customize logic functions by programming the device for specific applications. It provides up to twenty-nine array inputs and sixteen outputs. It incorporates AMD's unique input/output logic macrocell which provides flexible input/output structure and polarity, flexible feedback selection, multiple Output Enable choices, and a programmable clocking scheme. The macrocells can be individually programmed as "Combinatorial", "Registered", or "Latched" with active-HIGH or active-LOW polarity. The flexibility of the logic macrocells permits the system designer to tailor the device to particular application requirements.

Increased logic power has been built into the AmPAL29M16 by providing a variable number of logical product terms per output. Eight outputs have eight product terms each, four outputs have twelve product terms each, and the other four outputs have sixteen product terms each. This variable product-term distribution allows complex functions to be implemented in a single PAL device. Each output can be dynamically controlled by an Output Enable pin or Output Enable product terms. Each output can also be permanently enabled or diasabled.

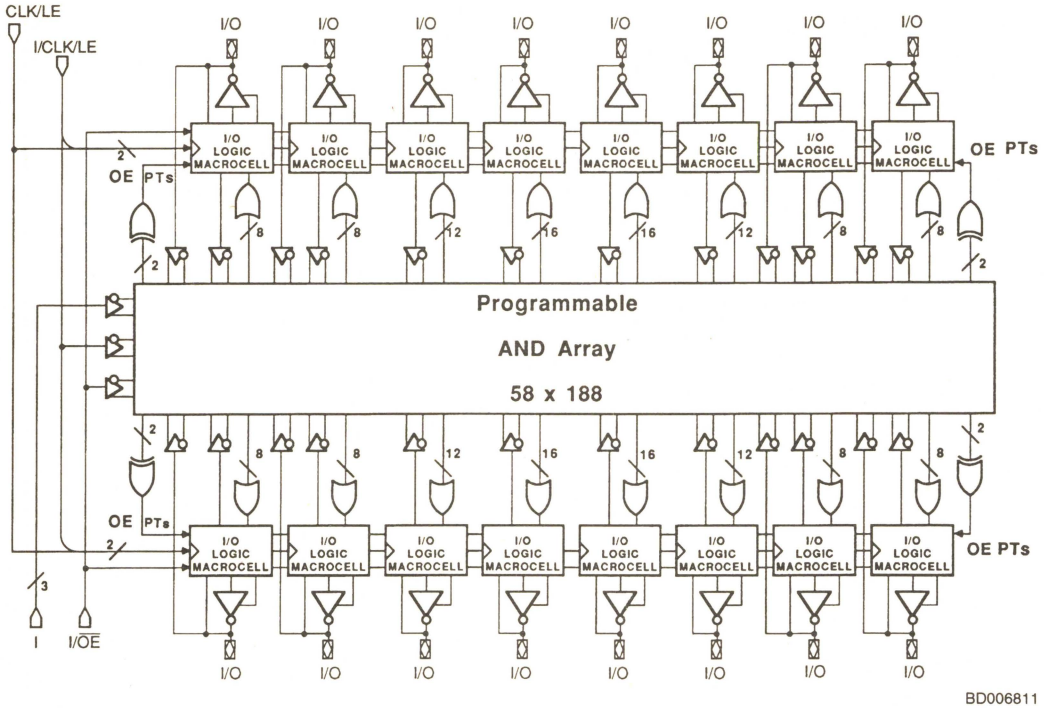
System operation has been enhanced by the addition of common asynchronous-PRESET and RESET product terms and a power-up RESET feature. The AmPAL29M16 also incorporates PRELOAD and Observability functions which permit full logical verification of the design.

The AmPALC29M16 is offered in the space-saving 300-Mil DIP package as well as chip carrier surface-mount packages.

5

08740B/0
JANUARY 1988

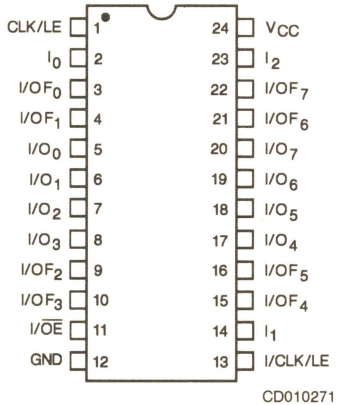
## Block Diagram



## Connection Diagrams

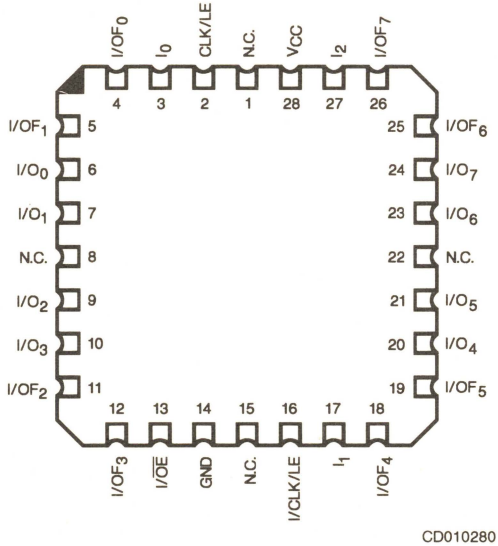
Top View

DIPs




CD010271

LCC*




CD010280

*Also available in PLCC. Pinouts identical to LCC.

Note: Pin 1 is marked for orientation.

5

## Pin Description

The following describes the functionality of all the pins on the 24-pin DIP. The 28-pin chip carrier has the same functionality with NO CONNECTS on pins 1,8,15,22.

**CLK/LE (PIN 1):**
Used as dedicated clock/latch enable pin for all registers/latches on the device if so selected. (See I/O Logic Macrocell Configurations.) This pin is a clock pin for macrocells configured as registers and a latch enable pin for macrocells configured as latches.

**I/CLK/LE PIN (PIN 13):**
Used as dedicated input or as an alternate clock/latch enable pin for all the registers/latches if so selected. (See I/O Logic Macrocell Configurations.) This pin is a clock pin for macrocells configured as registers and a latch enable pin for macrocells configured as latches.

**I/$\overline{OE}$ PIN (PIN 11):**
Used as a dedicated input pin to the AND array or as the Output Enable control pin (Active LOW) for all macrocells with pin-controlled Output Enable selected.

**$I_0$-$I_2$ (PINS 2,14,23):**
Dedicated input pins.

**$I/OF_0$-$I/OF_7$ (PINS 3,4,9,10,15,16,21,22):**
Eight bidirectional I/O pins with two independent feedback paths to the AND array. The first feedback path is a dedicated I/O pin feedback to the AND array for combinatorial input. The second feedback path consists of direct register/latch feedback to the array (see Figure 1).

**$I/O_0$-$I/O_7$ (PINS 5,6,7,8,17,18,19,20):**
Eight bidirectional I/O pins with user-programmable register/latch or I/O pin feedback to the AND array (see Figure 1).

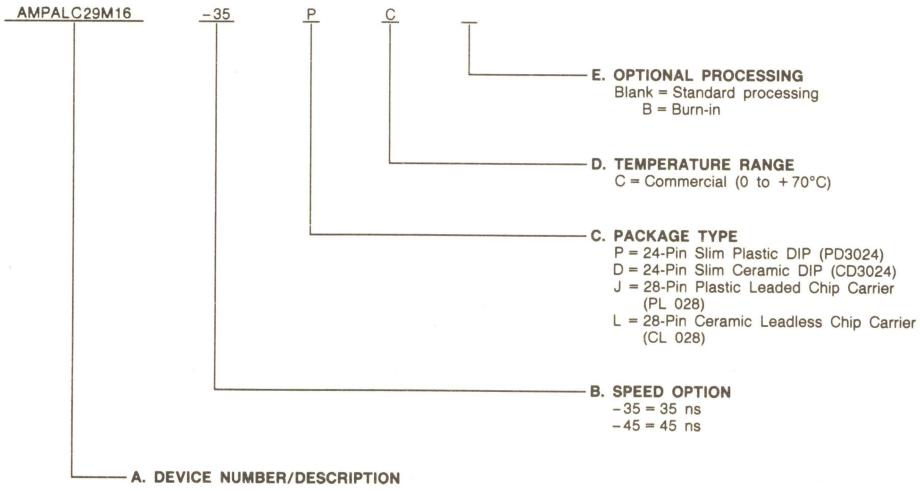
**$V_{CC}$ (PIN 24):**
Supply Voltage

**GND (PIN 12):**
Circuit Ground

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AmPALC29M16-35, -45 | PC, DC, DCB, JC, LC, LCB |

**Valid Combinations**

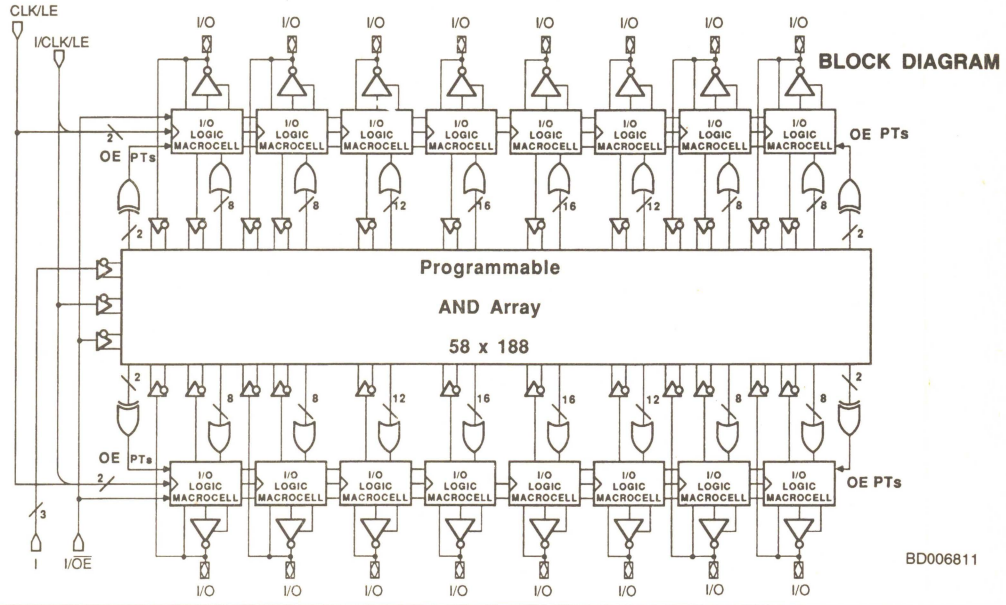
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Functional Description

### Inputs

The AmPAL29CM16 has 29 inputs to drive each product term (up to 58 inputs with both TRUE and complement versions available to the AND array) as shown in the block diagram below. Of these 29 inputs, 3 are dedicated inputs, 16 are from 8 I/O logic macrocells with 2 feedbacks, 8 are from other I/O logic macrocells with single feedback, 1 is for I/CLOCK/LE and 1 is for I/$\overline{OE}$ input.

Initially the AND-array gates are disconnected from all the inputs. This condition represents a logical TRUE to the AND array. By selectively programming the $E^2$cells, the AND array may be connected to either the TRUE input or the complement input. When both the TRUE and complement inputs are connected, a logical FALSE results at the output of the AND gate.




### Product Terms

The degree of programmability and complexity of a PAL device is determined by the number of connections that form the programmable-AND and OR gates. Each programmable-AND gate is called a product term. The AmPALC29M16 has 188 product terms. 176 of these product terms provide logic capability and 12 are architectural or control product terms. Among the 12 control product terms, 2 are for common Asynchronous-PRESET and RESET, 1 is for Observability, and 1 is for PRELOAD. The other 8 are common Output Enable product terms. The Output Enable of each bank of 4 macrocells can be programmed to be controlled by a common Output Enable pin or 2 AND/XOR product terms. It may be also permanently enabled or permanently disabled.

Each product term on the AmPALC29M16 consists of a 58-input AND gate. The outputs of these AND gates are connected to a fixed-OR plane. Product terms are allocated to OR gates in a variable distribution across the device ranging from 8-to-16 wide, with an average of 11 logical product terms per output. Increased number of product terms per output allows more complex functions to be implemented in a single PAL device. This flexibility aids in implementing functions such as counters, exclusive-OR functions, or complex state machines, where different states require different numbers of product terms.

Common asynchronous-PRESET and RESET product terms are connected to all Registered/Latched inputs/outputs. When the asynchronous-PRESET product term is asserted (HIGH) all the registers/latches will immediately be loaded with a HIGH, independent of the clock. When the asynchronous-RESET product term is asserted (HIGH) all the registers/latches will be immediately loaded with a LOW, independent of the clock. The actual output state will depend on the macrocell polarity selection. The latches must be in latched mode (not transparent mode) for the RESET/PRESET, PRELOAD, and power-up RESET modes to be meaningful.

### Input/Output Logic Macrocells

The I/O logic macrocell allows the user the flexibility of defining the architecture of each input or output on an individual basis. It also provides the capability of using the associated pin either as an input or an output.

The AmPALC29M16 has 16 macrocells, one for each I/O pin. Each I/O macrocell can be programmed for combinatorial, registered or latched operation (see Figure 1). Combinatorial output is desired when the PAL device is used to replace combinatorial glue logic. Registers are used in synchronous logic applications while latches are used in asynchronous applications where speed is critical. The output polarity for each macrocell in each of the three modes of operation is

5

user-selectable, allowing complete flexibility of the macrocell configuration.

Eight of the macrocells ($I/OF_0$-$I/OF_7$) have two independent feedback paths to the AND array (see Figure 1). The first is a dedicated I/O pin feedback to the AND array for combinatorial input. The second path consists of a direct register/latch feedback to the array. If the pin is used as a dedicated input using the first feedback path, the register/latch feedback path is still available to the AND array. This path provides the capability of using the register/latch as a buried state register/latch. The other eight macrocells have a single feedback path to the AND array. This feedback is user-selectable as either an I/O pin or a register/latch feedback.

Each macrocell can provide true input/output capability. The user can select each macrocell register/latch to be driven by either the output generated by the AND-OR array or the I/O pin. When the I/O pin is selected as the input, the feedback path provides the register/latch input to the array. When used as an input, each macrocell is also user-programmable for registered, latched, or combinatorial input.

The AmPALC29M16 has one dedicated CLK/LE pin and one I/CLK/LE pin. All macrocells have a programmable select to choose between these two pins as the clock or the latch enable signal. These pins are clock pins for macrocells configured as registers and latch enable pins for macrocells configured as latches. The polarity of these CLK/LE signals is also individually programmable. Thus different registers can be driven by multiple clocks and clock phases.

The Output-Enable mode of each of the macrocells can be selected by the user. The I/O pin can be configured as an output pin (permanently enabled) or as an input pin (permanently disabled). It can also be configured as a dynamic I/O controlled by the Output Enable pin or by two AND-XOR product terms which are available for each bank of four I/O logic macrocells.

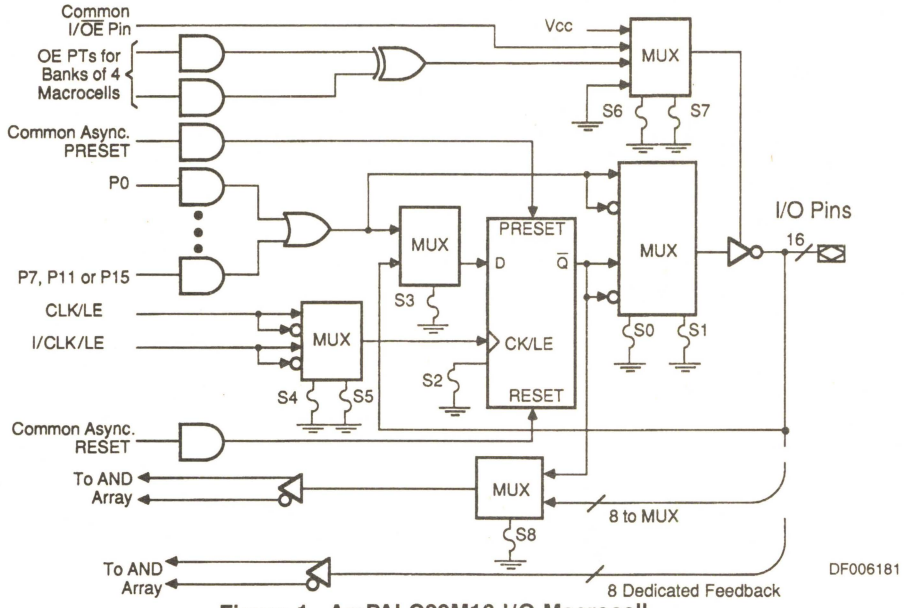



**Figure 1. AmPALC29M16 I/O Macrocell**

## I/O Logic Macrocell Configuration

AMD's unique I/O macrocell offers major benefits through its versatile, programmable input/output cell structure, multiple clock choices, flexible Output Enable and feedback selection. Eight I/O macrocells with single feedback contain nine $E^2$cells, while the other eight macrocells contain eight $E^2$cells for programming the input/output functions (see Table 1, Figure 2).

$E^2$cell S1 controls whether the macrocell will be combinatorial or registered/latched. S0 controls the output polarity (active-HIGH or active-LOW). S2 determines whether the output is a register or a latch. S3 allows the use of the macrocell as an input register/latch or as an output register/latch. It selects the direction of the data path through the register/latch. If connected to the usual AND-OR array output, the register/latch is an output connected to the I/O pin. If connected to the I/O pin, the register/latch becomes an input register/latch to the AND array using the feedback data path.

Programmable $E^2$cells S4 and S5 allow the user to select one of the four CLK/LE signals for each macrocell. S6 and S7 are used to control Output Enable as pin controlled, two product term controlled, permanently enabled or permanently disabled. S8 is a feedback multiplexer for the macrocells with a single feedback path only.

In the virgin erased state (charged, disconnected), an architectural cell is said to have a value of "1" ; In the programmed state (discharged, connected), an architectural cell is said to have a value of "0".

## Table 1. AmPALC29M16 I/O Logic Macrocell Architecture Selections

| S3 | I/O Cell |
|---|---|
| 1 | Output Cell |
| 0 | Input Cell |

| S2 | Storage Element |
|---|---|
| 1 | Register |
| 0 | Latch |

| S1 | Output Type |
|---|---|
| 1 | Combinatorial |
| 0 | Register/Latch |

| S0 | Output Polarity |
|---|---|
| 1 | Active LOW |
| 0 | Active HIGH |

| S8 | Feedback* |
|---|---|
| 1 | Register/Latch |
| 0 | I/O |

*Applies to macrocells with single feedback only.

TC003961

## Table 1. AmPALC29M16 I/O Logic Macrocell Clock Polarity & Output Enable Selections
(Cont'd.)

| S4 | S5 | Clock Edge/Latch Enable Level |
|---|---|---|
| 1 | 1 | CLK/LE pin positive-going edge, active-HIGH LE |
| 1 | 0 | CLK/LE pin negative-going edge, active-LOW LE |
| 0 | 1 | I/CLK/LE pin positive-going edge, active-HIGH LE |
| 0 | 0 | I/CLK/LE pin negative-going edge, active-LOW LE |

| S6 | S7 | Output Buffer Control |
|---|---|---|
| 1 | 1 | Pin-Controlled 3-State Enable |
| 1 | 0 | XOR PT-Controlled 3-State Enable |
| 0 | 1 | Permanently Enabled (Output only) |
| 0 | 0 | Permanently Disabled (Input only) |

TC003971

1 = Erased State (Charged or disconnected)
0 = Programmed State (Discharged or connected)

5

## Some Possible Configurations of the Input/Output Logic Macrocell

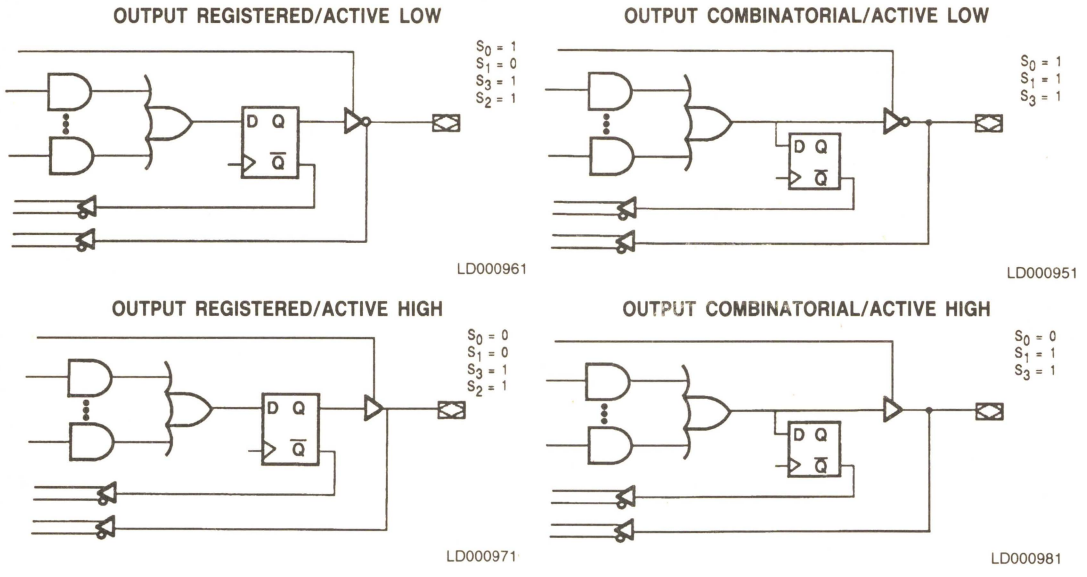



Figure 2a. Dual Feedback Macrocells

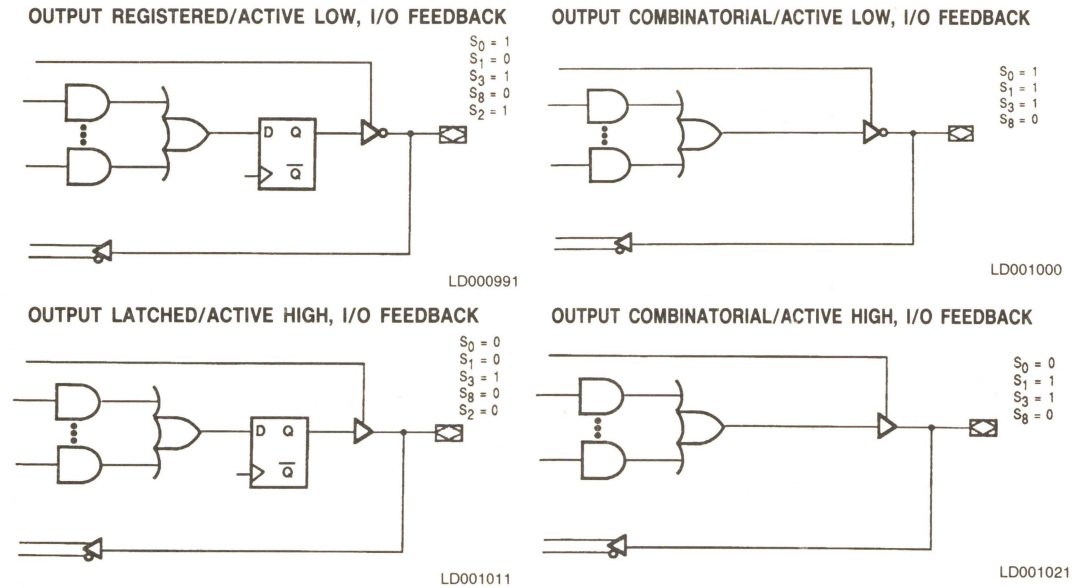
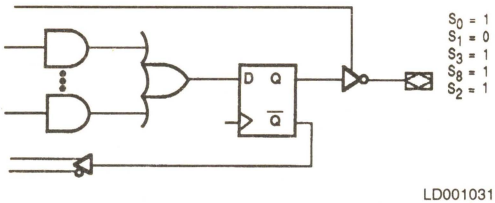
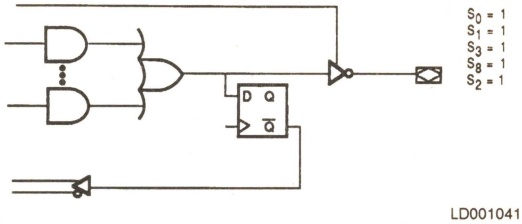



Figure 2b. Single Feedback Macrocells

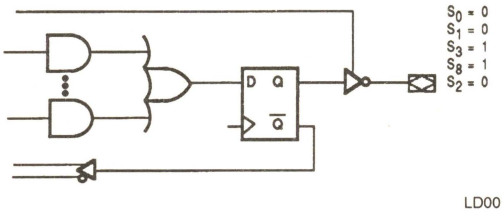
OUTPUT REGISTERED/ACTIVE LOW, REG. FEEDBACK




OUTPUT COMBINATORIAL/ACTIVE LOW, REG. FEEDBACK




OUTPUT LATCHED/ACTIVE LOW, LATCHED FEEDBACK




OUTPUT COMBINATORIAL/ACTIVE LOW, LATCH FEEDBACK

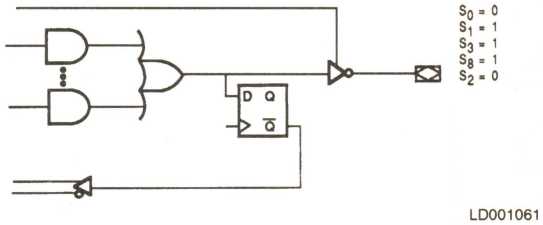



**Figure 2b. Single Feedback Macrocells** (Cont'd.)

INPUT REGISTERED/LATCHED

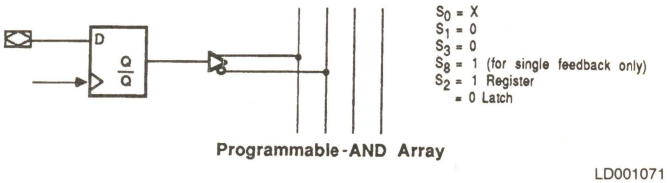



**Figure 2c. All Macrocells**

5

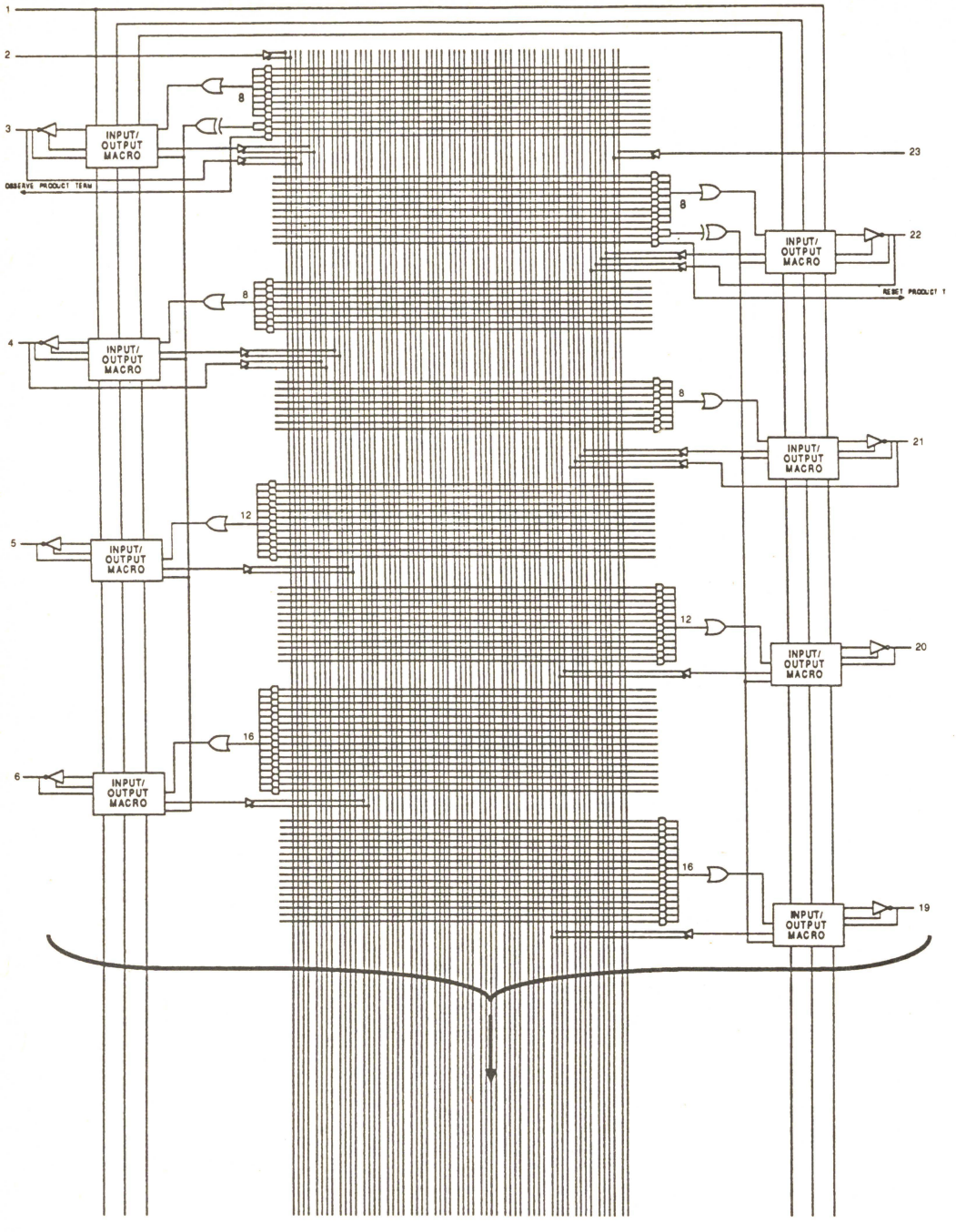



Figure 3. AmPALC29M16 Logic Diagram

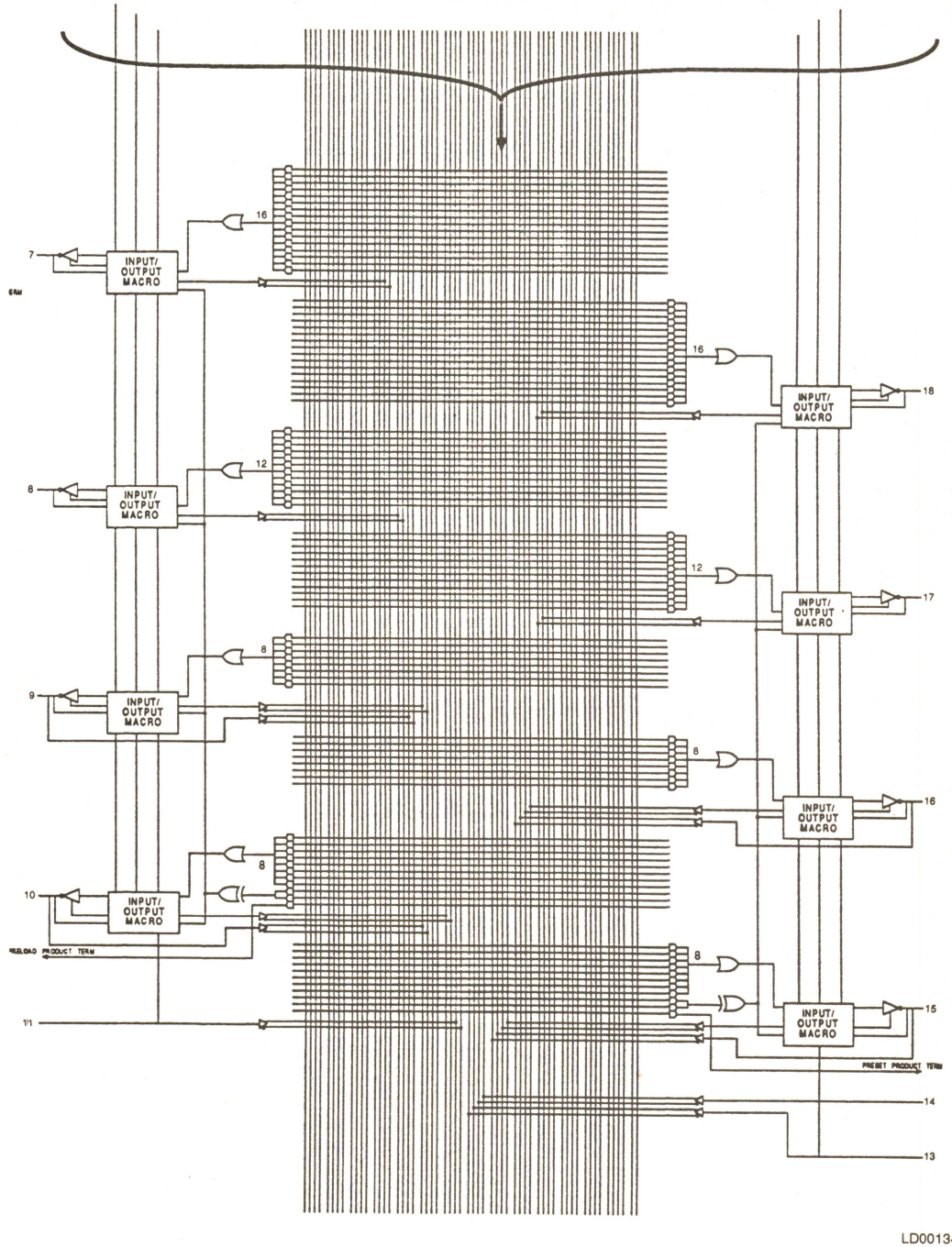



**Figure 3. AmPALC29M16 Logic Diagram** (Cont'd.)

## Designed in Testability and Debugging

### PRELOAD

To simplify testing, the AmPALC29M16 is designed with PRELOAD circuitry that provides an easy method for testing logical functionality. Both product-term controlled and supervoltage-enabled PRELOAD modes are available. This offers even more test capability than previously implemented in AMD's PAL devices. The TTL-level PRELOAD product term can be useful during debugging, where supervoltages may not be available.

PRELOAD allows any arbitrary state value to be loaded into the registers/latches of the device. A typical functional-test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the devices inputs into any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state, or "next state", which can be checked to validate the transition from the "present state". In this way any transition can be checked.

Since PRELOAD can provide the capability to go directly to any desired arbitrary state, test sequences may be greatly shortened. Also, all possible states can be tested, thus greatly reducing test time and development costs and guaranteeing proper in-system operation.

### Observability

The output register/latch observability product term, when asserted, suppresses the combinatorial output data from appearing on the I/O pin and allows the observation of the contents of the register/latch on the output pin for each of the logic macrocells. This unique feature allows for easy debugging and tracing of the buried state machines. In addition, a capability of supervoltage observability is also provided.

### Power-Up Reset

All the device registers/latches have been designed to reset during device power-up. Following the power-up, all registers/latches will be cleared (Q = 0), setting the outputs to a state determined by the output select multiplexer. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

### Security Cell

A security cell is provided on each device to prevent unauthorized copying of the user's proprietary logic design. Once programmed, the security cell disables the programming, verification, PRELOAD, and the observability modes. The only way to erase the protection cell is by charging the entire array and architecture cells, in which case no proprietary design can be copied. (This cell should be programmed only after the rest of the device has been completely programmed and verified.)

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Ambient Temperature under bias..... −55 to +125°C
Supply Voltage with
Respect to Ground..... −0.5 V to +7.0 V
DC Output Voltage..... −0.5 V to $V_{CC}$ + 0.5 V
DC Input Voltage
(Except Pin I/OE) ..... −0.5 V to $V_{CC}$ + 0.5 V
DC Input Voltage (Pin I/OE) ..... −0.6 V to +17 V
DC Input Current ..... −1 mA to +1 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$)..... 0°C to +70°C
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Consult Factory for Military Specifications

## DC Characteristics over operating range unless otherwise specified

### HCT Devices*

| Parameter Symbol | Parameter Description | Test Conditions | | Min. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min.<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$<br>$I_{OH}$ = −2 mA | | 2.4 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min.<br>$V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 6 mA | | 0.5 | V |
| | | | $I_{OL}$ = 4 mA | | 0.33 | |
| | | | $I_{OL}$ = 20 μA | | 0.1 | |
| $V_{IH}$ | Input HIGH Voltage | Guaranteed Logic HIGH for all Inputs | | 2.0 | | V |
| $V_{IL}$ | Input LOW Voltage | Guaranteed Logic LOW for all Inputs | | | 0.8 | V |
| $I_I$ | Input Leakage Current | $V_{IN}$ = 0 to 5.5 V, $V_{CC}$ = Max. | | | 10 | μA |
| $I_O$ | Output Leakage Current | $V_{IN}$ = 0 to 5.5 V, $V_{CC}$ = Max. | | | 10 | μA |
| $I_{CCOP}$ | Operating Current Supply | f = $f_{MAX}$, Outputs Open ($I_O$ = 0) | | | 120 | mA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_O$ = 0 V | | −30 | −90 | mA |

### Capacitance

| Parameter Symbol | Parameter Description | Test Conditions | Typ. | Units |
|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{CC}$ = 5.00 V., $T_A$ = 25°C<br>$V_{IN}$ = 0 V @ f = 1 MHz | 5 | pF |
| $C_{OUT}$ | Output Capacitance | | 8 | |

Note: These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

* Consult factory for DC specification on HC Devices.

5

## Switching Characteristics

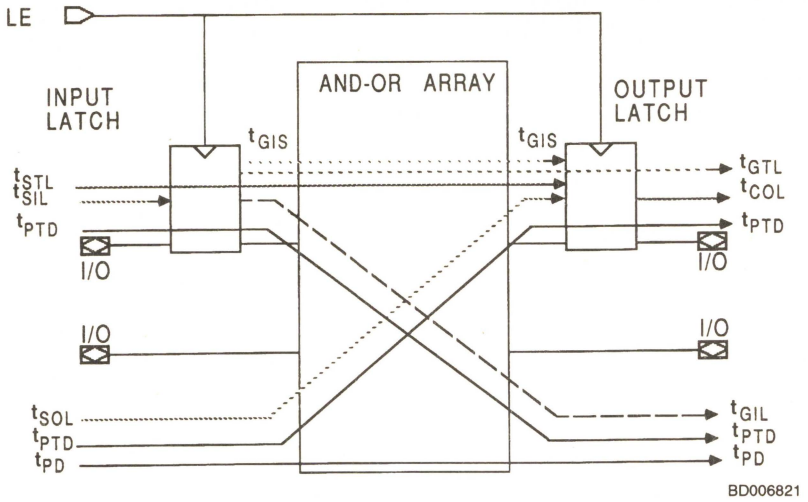
over operating range unless otherwise specified; all values are determined under the loading of one TTL gate and a capacitance of 50 pF

| Parameter Number | Parameter Symbol | Parameter Description | -35 | | -45 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| **REGISTERED OPERATION (Numbers 1 through 12)** | | | | | | | |
| 1 | $t_{PD}$ | Input or I/O Pin to Combinatorial Output | | 35 | | 45 | ns |
| **Output Register** | | | | | | | |
| 2 | $t_{SOR}$ | Input or I/O Pin to Output Register Setup | 27 | | 34 | | ns |
| 3 | $t_{COR}$ | Output Register Clock to Output | | 23 | | 32 | ns |
| 4 | $t_{HOR}$ | Data Hold Time for Output Register | 0 | | 0 | | ns |
| **Input Register** | | | | | | | |
| 5 | $t_{SIR}$ | I/O Pin to Input Register Setup | 6 | | 8 | | ns |
| 6 | $t_{CIR}$ | Register Feedback Clock to Combinatorial Output | | 45 | | 58 | ns |
| 7 | $t_{HIR}$ | Data Hold Time for Input Register | 3 | | 4 | | ns |
| **Clocking and Frequency** | | | | | | | |
| 8 | $t_{CIS}$ | Register Feedback to Output Register/Latch Setup | 35 | | 45 | | ns |
| 9 | $f_{MAX}$ | Maximum Frequency $1/(t_{SOR} + t_{COR})$ | | 20 | | 15 | MHz |
| 10 | $f_{MAXI}$ | Max Internal Frequency $1/t_{CIS}$ | | 28.5 | | 22.5 | MHz |
| 11 | $t_{CWH}$ | Clock Width HIGH | 12 | | 15 | | |
| 12 | $t_{CWL}$ | Clock Width LOW | 12 | | 15 | | |
| **LATCH OPERATION (Numbers 13 through 24)** | | | | | | | |
| 13 | $t_{PD}$ | Input or I/O Pin to Combinatorial Output | | 35 | | 45 | ns |
| 14 | $t_{PTD}$ | Input or I/O Pin to Output via One Transparent Latch | | 45 | | 55 | ns |
| **Output Latch** | | | | | | | |
| 15 | $t_{SOL}$ | Input or I/O Pin to Output Latch Setup | 27 | | 34 | | ns |
| 16 | $t_{GOL}$ | Latch Enable to Transparent Mode Output | | 23 | | 32 | ns |
| 17 | $t_{HOL}$ | Data Hold Time for Output Latch | 0 | | 0 | | ns |
| 18 | $t_{STL}$ | Input or I/O Pin to Output Latch Setup via Transparent Input Latch | 35 | | 45 | | ns |

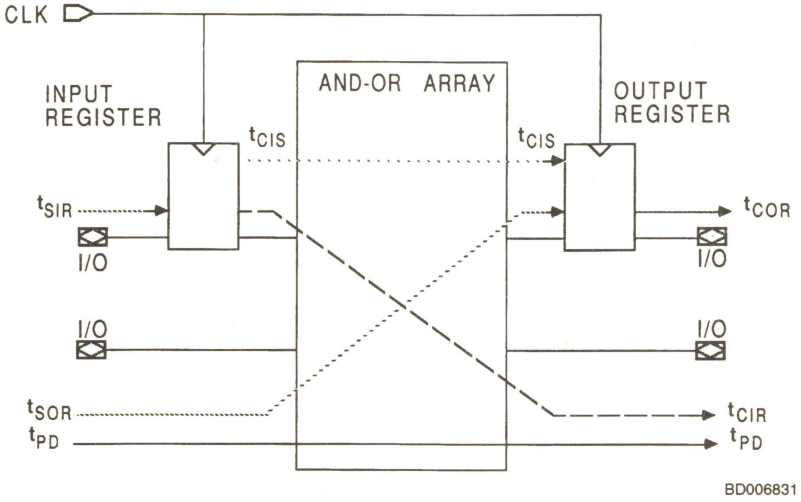
| Parameter Number | Parameter Symbol | Parameter Description | -35 | | -45 | | Units |
|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | |
| **Input Latch** | | | | | | | |
| 19 | $t_{SIL}$ | I/O Pin to Input Latch Setup | 6 | | 8 | | ns |
| 20 | $t_{GIL}$ | Latch Feedback, Latch Enable Transparent Mode to Combinatorial Output | | 45 | | 58 | ns |
| 21 | $t_{HIL}$ | Data Hold Time for Input Latch | 3 | | 4 | | ns |
| **Latch Enable** | | | | | | | |
| 22 | $t_{GIS}$ | Latch Feedback Transparent Mode to Output Register/ Latch Setup | 35 | | 45 | | ns |
| 23 | $t_{GWH}$ | Latch Enable Width HIGH | 12 | | 15 | | ns |
| 24 | $t_{GWL}$ | Latch Enable Width LOW | 12 | | 15 | | ns |
| **RESET/PRESET & OUTPUT ENABLE OPERATION (Numbers 25 through 32)** | | | | | | | |
| 25 | $t_{APO}$ | Input or I/O Pin to Output Register/Latch RESET/ PRESET | | 40 | | 55 | ns |
| 26 | $t_{AW}$ | Async. RESET/PRESET Pulse Width | 35 | | 45 | | ns |
| 27 | $t_{ARO}$ | Async. RESET/PRESET to Input Register/Latch Recovery | 30 | | 40 | | ns |
| 28 | $t_{ARI}$ | Async. RESET/PRESET to Input Register/Latch Recovery | 20 | | 30 | | |
| 29 | $t_{PZX}$ | I/$\overline{OE}$ Pin to Output Enable | | 30 | | 40 | ns |
| 30 | $t_{PXZ}$ * | I/$\overline{OE}$ Pin to Output Disable | | 30 | | 40 | ns |
| 31 | $t_{EA}$ | Input or I/O to Output Enable via PT | | 35 | | 45 | ns |
| 32 | $t_{ER}$ * | Input or I/O to Output Disable via PT | | 35 | | 45 | ns |

* Output disable times do not include test load RC time constants.

5

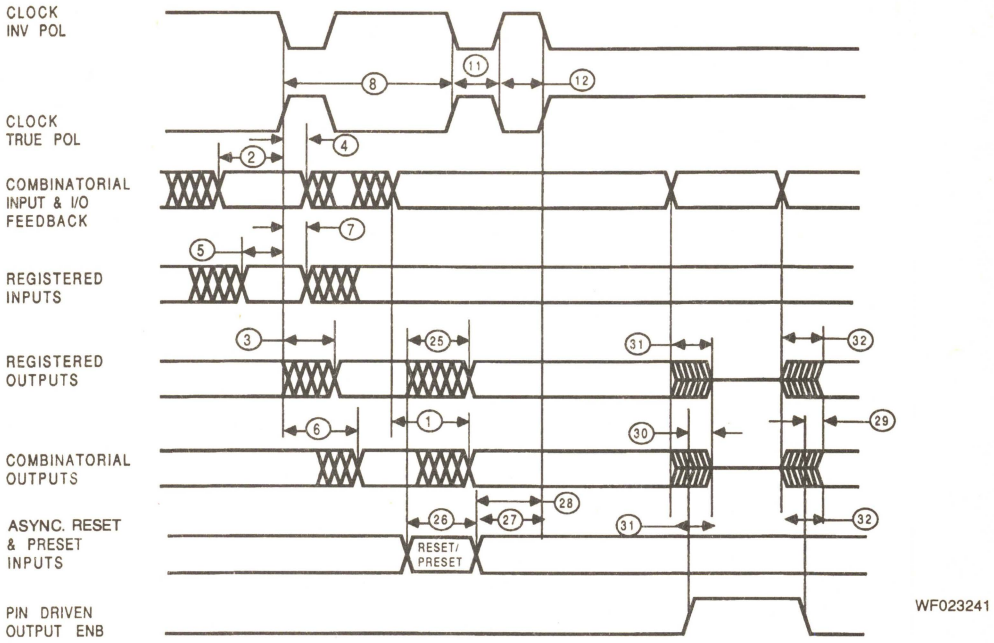



**Input/Output Specs (Pin LE Reference)**

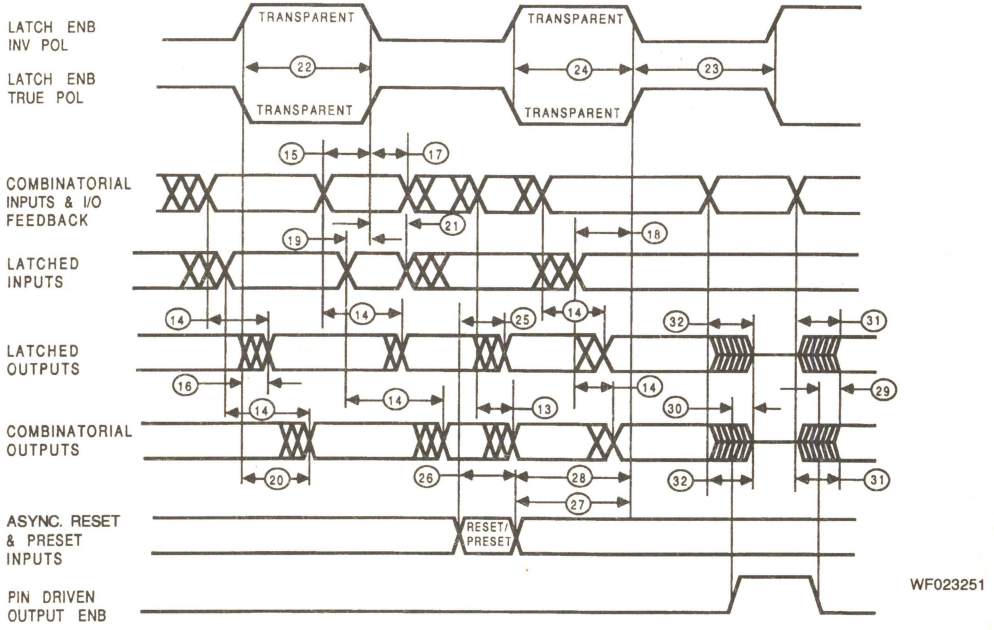



**Input/Output Register Specs (Pin CLK Reference)**

## Switching Waveforms

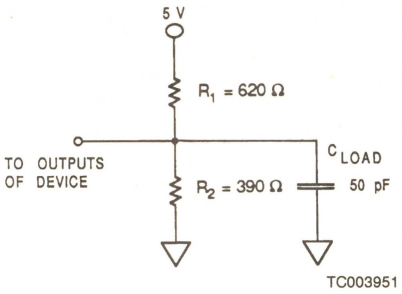



**Register**

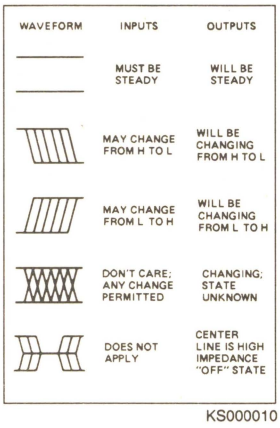



**Latch**

## Switching Test Circuit




## Key to Switching Waveforms



# AmPAL22V10-15

24-Pin IMOX III™ Programmable Array Logic

PRELIMINARY

## Distinctive Characteristics

- 15-ns performance
- Increased logic power — up to 22 inputs and 10 outputs
- Increased product terms — average 12 per output
- Variable product-term distribution improves ease of use
- Each output user-programmable for registered or combinatorial operation
- Individually user-programmable output polarity
- Extra terms provide logical synchronous-PRESET and asynchronous-RESET capability
- TTL-level PRELOAD for improved testability
- Packaged in 24-pin Slim DIP and 28-pin chip-carrier packages
- Platinum-Silicide fuses ensure high programming yield, fast programming, and high reliability
- AC and DC testing done at the factory utilizing special designed-in test features
- 3000-V Input ESD Protection on all pins

## General Description

The AmPAL22V10 is a second-generation Programmable Array Logic (PAL) device. It utilizes the familiar sum-of-products (AND-OR) logic structure, allowing users to program custom logic functions. The AmPAL22V10 permits the development of custom LSI functions of 500 to 800 equivalent gate complexity.

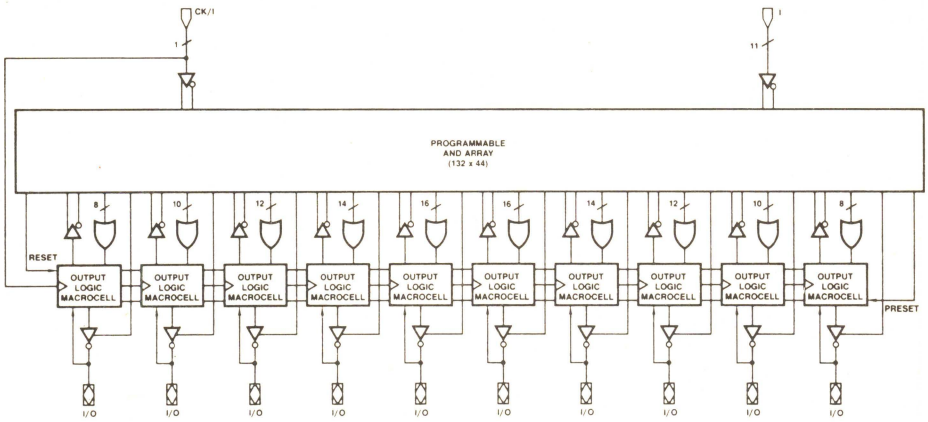
The AmPAL22V10 contains up to 22 inputs and 10 outputs. It incorporates the unique capability of defining and programming the architecture of each output on an individual basis. Each output is user-programmable for either registered or combinatorial operation. This allows the designer to optimize the device design by having only as many registers as needed. In addition, each output has user-programmable output polarity, further simplifying design and contributing to the precise application requirements.

Increased logic power has been built into the AmPAL22V10 by increasing the number of product terms from 8-per-output to an average of 12-per-output. Further innovation can be seen in the introduction of variable product-term distribution. This technique allocates from 8 to 16 logical product terms to each output (please refer to the Block Diagram for distribution details). This variable allocation of terms allows far more complex functions to be implemented than in previous devices.

System operation has been enhanced by the addition of a synchronous-PRESET and an asynchronous-RESET product term. These terms are common to all output registers.

The AmPAL22V10 also incorporates power-up RESET and the capability to PRELOAD the output registers to any desired state during testing. PRELOAD is essential to permit full logical verification during test.

## Block Diagram

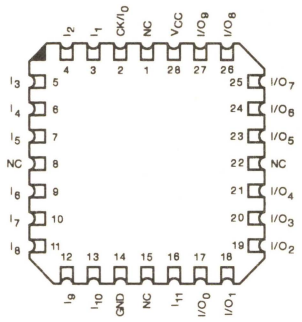



BD006590

## Connection Diagram

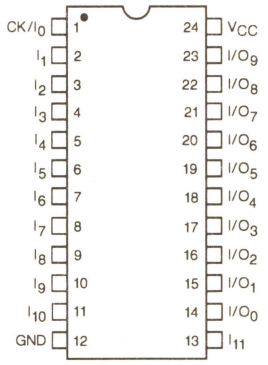
Top View

LCC*




CD010001

DIPs**




CD010011

*Also available in PLCC. Pinouts identical to LCC.

**Also available in Flatpack. Pinouts identical to DIPs.

**Pin Designations:** I = Input
I/O = Input/Output
$V_{CC}$ = Supply Voltage
GND = Ground
CK = Clock
NC = No Connection

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

**a. Device Number**
**b. Speed/Power Option** (if applicable)
**c. Package Type**
**d. Temperature Range**
**e. Optional Processing**

AMPAL22V10 -15 P C —

**e. OPTIONAL PROCESSING**
Blank = Standard processing
B = Burn-in

**d. TEMPERATURE RANGE**
C = Commercial (0 to +75°C)
E = Extended Commercial (−55 to +125°C)

**c. PACKAGE TYPE**
P = 24-Pin Slim Plastic DIP (PD3024)
D = 24-Pin Slim Ceramic DIP (CD3024)
J = 28-Pin Plastic Leaded Chip Carrier (PL 028)
L = 28-Pin Ceramic Leadless Chip Carrier (CL 028)

**b. SPEED/POWER OPTION**
−15 = 15 ns (Full Power)

**a. DEVICE NUMBER/DESCRIPTION**
AmPAL22V10-15
24-Pin IMOX III Programmable Array Logic

| Valid Combinations | |
|---|---|
| AMPAL22V10-15 | PC, DC, DCB, DE, JC, LC, LE |

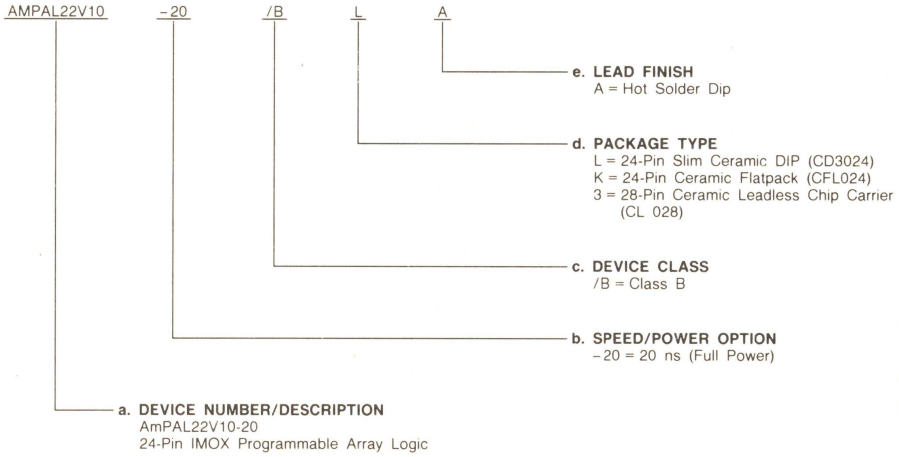
**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

# Military Ordering Information

## APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of: **a. Device Number**
**b. Speed/Power Option** (if applicable)
**c. Device Class**
**d. Package Type**
**e. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL22V10-20 | /BLA/B3A/BKA |

### Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

### Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, and 11.

5

## Functional Description

The AmPAL22V10 is a second-generation Programmable Array Logic device. It contains a programmable fuse array organized in the familiar sum-of-products (AND-OR) structure.

The block diagram below shows the basic architecture of the AmPAL22V10. There are up to 22 inputs and 10 outputs available. The inputs are connected to a programmable-AND array which contains 120 logical product terms. Initially the AND gates are connected, via fuses, to both the TRUE and COMPLEMENT of every input. By selective programming of fuses, the AND gates may be "connected" to only the TRUE input (by programming the COMPLEMENT fuse), to only the COMPLEMENT input (by programming the TRUE fuse), or to neither type of input (by programming both fuses), establishing a logical "don't care." When both the TRUE and COMPLEMENT fuses are left intact, a logical FALSE results on the output of the AND gate. An AND gate with all fuses programmed will assume the logical-TRUE state. The outputs of the AND gates are connected to fixed-OR gates. There is an average of 12 product terms per OR gate (output), and as the Block Diagram shows, variable product term distribution has been implemented. This technique allocates different quantities of logical product terms to different outputs, allowing more complex logical functions to be performed than were previously possible. Up to 16 logical terms can be evaluated in one output in a single clock cycle (no feedback necessary).

### Output Logic Macrocells (OLMs)

A dramatic innovation in logic design is the implementation on the AmPAL22V10 of variable output architecture. This allows the user to program, on an output-by-output basis, the function of the outputs. As shown in the Output Logic Macrocell (OLM) diagram below, each output cell contains two additional fuses, $S_0$ and $S_1$. $S_1$ controls whether the output will be registered or combinatorial. $S_0$ controls the output polarity (active HIGH or active LOW). Depending on the states of these two fuses, an individual output will operate in one of four modes (see the logic diagrams on the next page): Registered/Active LOW, Registered/Active HIGH, Combinatorial/Active LOW, and Combinatorial/Active HIGH (note that the feedback path also changes with output mode). This innovation gives the designer more flexibility and enables optimization of the device for precise application requirements. It also allows for better device utilization — programming only as many registers as are needed.

### PRESET/RESET

To improve in-system functionality, the AmPAL22V10 has additional PRESET and RESET product terms. These terms are connected to all registered outputs. When the synchronous-PRESET product term is asserted (HIGH), the output registers will be loaded with a HIGH on the next LOW-to-HIGH clock transition. When the asynchronous-RESET product term is asserted (HIGH), the output registers will be immediately loaded with a LOW (independent of the clock). These functions are particularly useful for applications such as system power-on and reset.

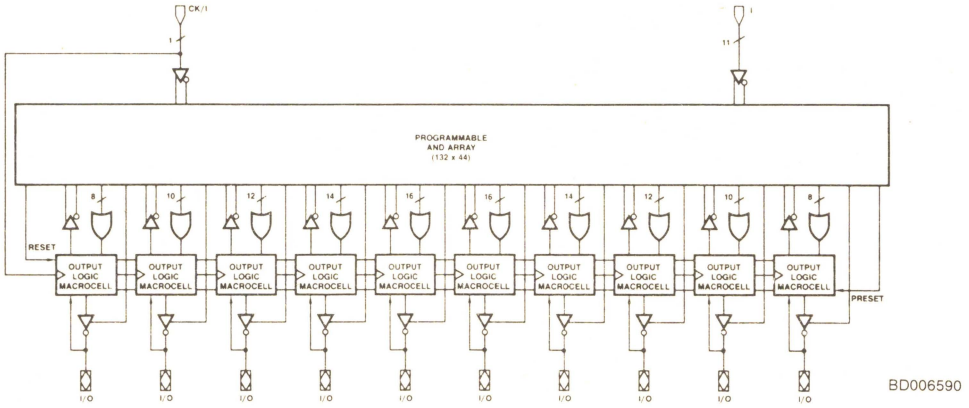



Figure 1. Block Diagram

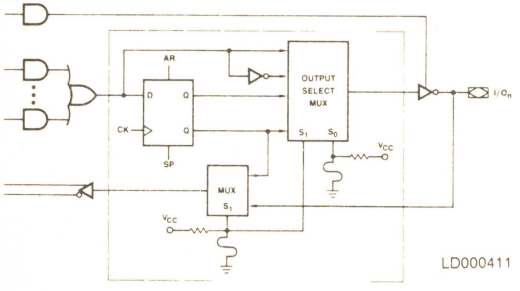



Figure 2. Output Logic Macrocell Diagram

| $S_1$ | $S_0$ | Output Configuration |
|---|---|---|
| 0 | 0 | Registered/Active LOW |
| 0 | 1 | Registered/Active HIGH |
| 1 | 0 | Combinatorial/Active LOW |
| 1 | 1 | Combinatorial/Active HIGH |

0 = Unblown Fuse
1 = Blown Fuse

## PRELOAD

To simplify testing, the AmPAL22V10 is designed with PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the output registers.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. To verify these transitions requires the ability to set the state registers into an arbitrary "present state" value, and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is then clocked into a new state, or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

## Fabrication

The AmPAL22V10-15 is manufactured using Advanced Micro Devices' IMOX III slot-isolation process. This advanced process is an extension of the IMOX II process. It uses slot isolation between transistors to permit an increase in density and a decrease in internal capacitance, resulting in the fastest possible programmable-logic devices.

These devices are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are preprogrammed during manufacturing to ensure extremely high field programming yields ( > 98.5%), and to provide extra test paths to achieve excellent parametric correlation.

Platinum Silicide was selected as the fuse-link material to achieve a well-controlled melt rate resulting in large nonconductive gaps that ensure very stable, long-term reliability. Extensive operating testing has proven that this low-field, large-gap technology offers high reliability for fusible-link programmable logic.

## Security Fuse Programming

A single fuse is provided on each AmPAL22V10 part to prevent unauthorized copying of PAL fuse patterns. Once programmed, the circuitry enabling fuse verification and registered output PRELOAD is permanently disabled.

Programming of the security fuse is the same as an array fuse. Verification of a programmed security fuse is accomplished by verifying the whole fuse array as if every fuse were programmed.

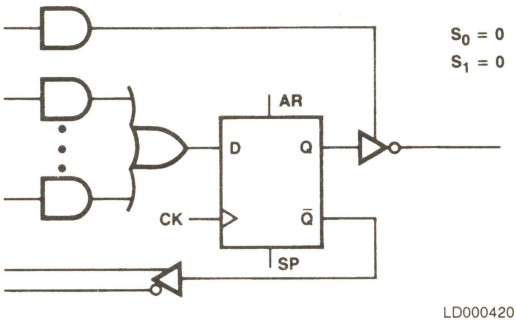



Figure 3-1. Registered/Active LOW

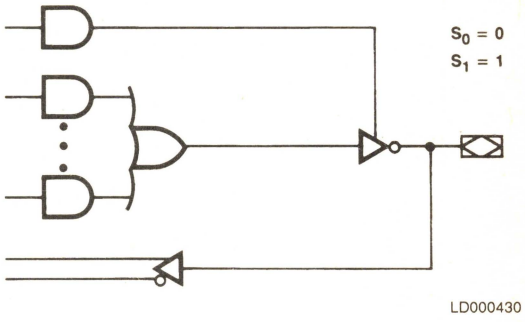



Figure 3-3. Combinatorial/Active LOW

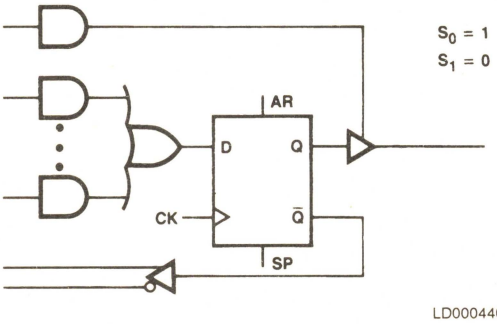



Figure 3-2. Registered/Active HIGH

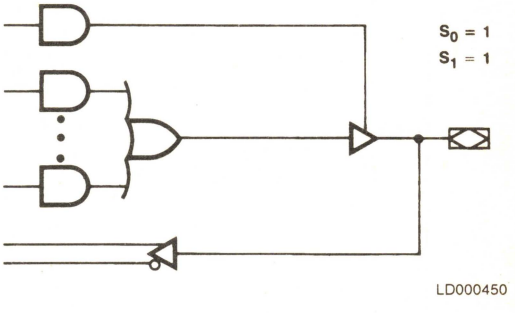



Figure 3-4 Combinatorial/Active HIGH

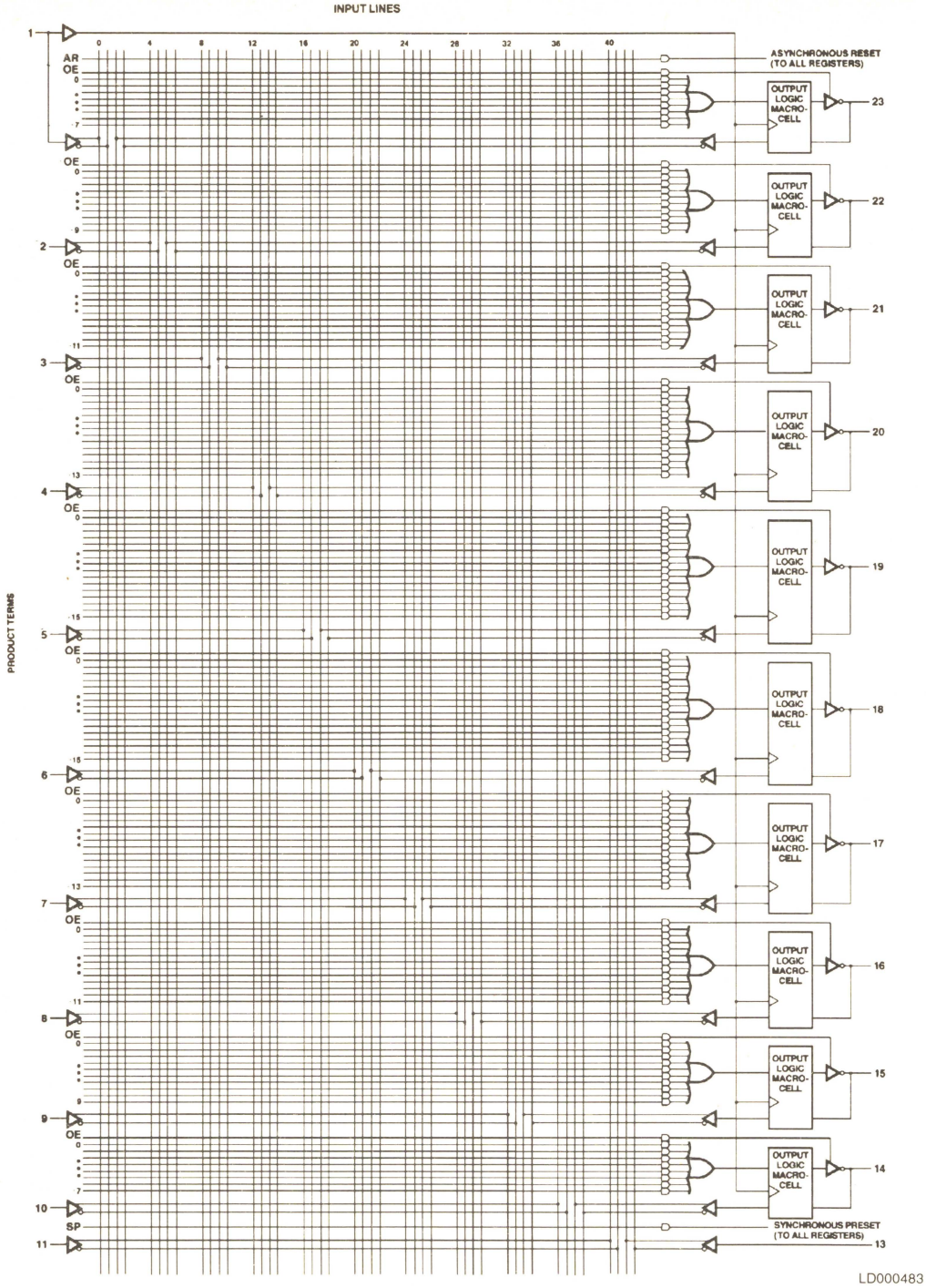



Figure 4. Logic Diagram

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
(Pin 24 to Pin 12) Continuous ..... −0.5 to +7 V
DC Voltage Applied to Outputs
(Except During Programming) ..... −0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 16 V
Output Current into Outputs During Programming
(Max. Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5 mA
Ambient Temperature with Power Applied ..... +125°C

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) Operating Free Air ..... 0°C to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) Operating Case ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military Product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

**DC CHARACTERISTICS** over operating range unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

| Parameter Symbol | Parameter Description | Test Conditions | COM'L | MIL & E-COM'L | Min. | Typ. (Note 1) | Max. | Unit |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | $I_{OH}$ = −2 mA | 2.4 | 3.5 | | V |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 16 mA | $I_{OL}$ = 12 mA | | | 0.50 | V |
| $V_{IH}$ (Note 2) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for all Inputs | | | 2.0 | | | V |
| $V_{IL}$ (Note 2) | Input LOW Level | Guaranteed Input Logical LOW Voltage for all Inputs | | | | | 0.8 | V |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | −20 | −100 | μA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | 25 | μA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | −30 | −50 | −90 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | | B | | 150 | 180 | mA |
| | | | | AL | | 75 | 90 | |
| | | | | Q | | 40 | 55 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | −0.9 | −1.2 | V |
| $I_{OZH}$ | Output Leakage Current (Note 4) | $V_{CC}$ = Max., $V_{IN}$ = $V_{IL}$ or $V_{IH}$ | $V_O$ = 2.7 V | | | | 100 | μA |
| $I_{OZL}$ | | | $V_O$ = 0.4 V | | | | −100 | |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground, and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short-circuit test should not exceed one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).
5. Pinout for DIPs only.

## CAPACITANCE

| Parameter Symbol | Parameter Description | Test Conditions | | Min. | Typ. (Note 1) | Max. | Unit |
|---|---|---|---|---|---|---|---|
| $C_{IN}$ † | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz (Note 5) | Pins 1, 13 | | 9 | | pF |
| | | | Others | | 6 | | |
| $C_{OUT}$ † | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | | | 9 | | |

† Not included in Group A tests.

## Switching Characteristics Over Commercial Operating Range (Note 1)

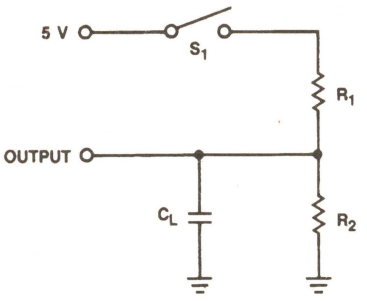
| No. | Parameter Symbol | Parameter Description | 22V10-15 | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output | | 15 | | | | | ns |
| 2 | $t_{EA}$ | Input to Output Enable | | 15 | | | | | ns |
| 3 | $t_{ER}$ | Input to Output Disable | | 15 | | | | | ns |
| 4 | $t_{CO1}$ | Clock to Output | | 12 | | | | | ns |
| 5 | $t_{CO2}$ | Register Feedback through Array to Combinatorial Output, Relative to External Clock | | 22 | | | | | ns |
| 6 | $t_S$ | Input or Feedback Setup Time | 13 | | | | | | ns |
| 7 | $t_H$ | Hold Time | 0 | | | | | | ns |
| 8 | $t_P$ | Clock Period ($t_S + t_{CO1}$) | 25 | | | | | | ns |
| 9 | $t_{WH}$ | Clock Width – HIGH Level | 10 | | | | | | ns |
| 10 | $t_{WL}$ | Clock Width – LOW Level | 10 | | | | | | ns |
| 11 | $f_{MAX}$ | Maximum Frequency | | 40 | | | | | ns |
| 12 | $t_{AW}$ | Asynchronous – RESET Width | 15 | | | | | | ns |
| 13 | $t_{AR}$ | Asynchronous – RESET Recovery Time | 15 | | | | | | ns |
| 14 | $t_{AP}$ | Asynchronous – RESET to Registered Output | | 20 | | | | | ns |

## Switching Characteristics Over Military and Extended-Commercial Operating Ranges
**(for APL Products, Group A, Subgroups 7, 8, 9, 10, 11 are tested unless otherwise noted) (Note 2)**

| No. | Parameter Symbol | Parameter Description | 22V10-20 | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Max. | Min. | Max. | Min. | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output | | 20 | | | | | ns |
| 2 | $t_{EA}$ | Input to Output Enable | | 20 | | | | | ns |
| 3 | $t_{ER}$ | Input to Output Disable | | 20 | | | | | ns |
| 4 | $t_{CO1}$ | Clock to Output | | 15 | | | | | ns |
| 5 | $t_{CO2}$ | Register Feedback through Array to Combinatorial Output, Relative to External Clock | | 30 | | | | | ns |
| 6 | $t_S$ | Input or Feedback Setup Time | 17 | | | | | | ns |
| 7 | $t_H$ | Hold Time | 0 | | | | | | ns |
| 8 | $t_P$ | Clock Period ($t_S + t_{CO1}$) | 32 | | | | | | ns |
| 9 | $t_{WH}$ | Clock Width – HIGH Level | 15 | | | | | | ns |
| 10 | $t_{WL}$ | Clock Width – LOW Level | 15 | | | | | | ns |
| 11 | $f_{MAX}$ | Maximum Frequency | | 31 | | | | | ns |
| 12 | $t_{AW}$ | Asynchronous – RESET Width | 20 | | | | | | ns |
| 13 | $t_{AR}$ | Asynchronous – RESET Recovery Time | 20 | | | | | | ns |
| 14 | $t_{AP}$ | Asynchronous – RESET to Registered Output | | 25 | | | | | ns |

Notes: 1. Commercial Test Conditions: $R_1 = 300$, $R_2 = 390$.
2. Military/Extended-Commercial Test Conditions: $R_1 = 390$, $R_2 = 750$.
3. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
4. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high-impedance to HIGH tests and closed for high-impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high-impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high-impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

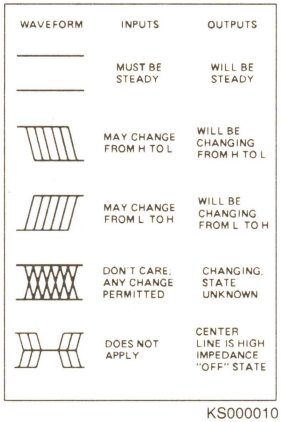
## Switching Test Circuit




TC001190

## Switching Waveforms

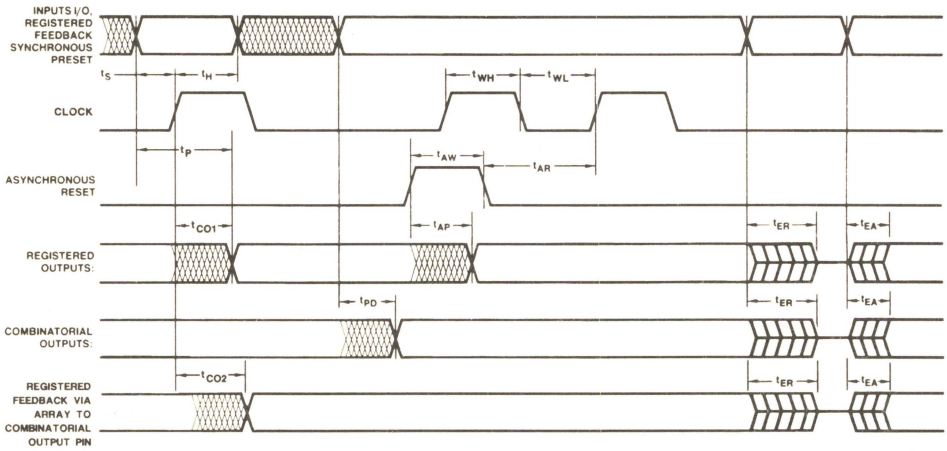
KEY TO SWITCHING WAVEFORMS




KS000010

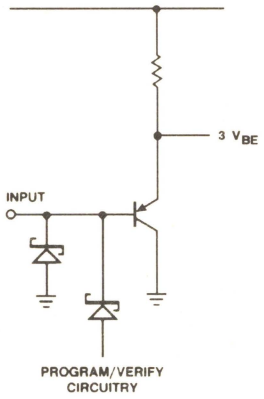
5

## Switching Waveforms (Continued)



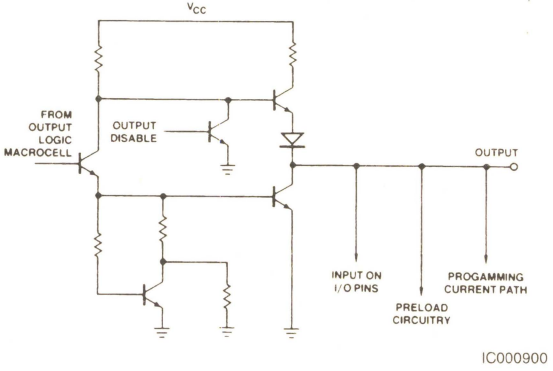

WF022287

## Input/Output Circuit Diagram




IC000893

Input Circuitry



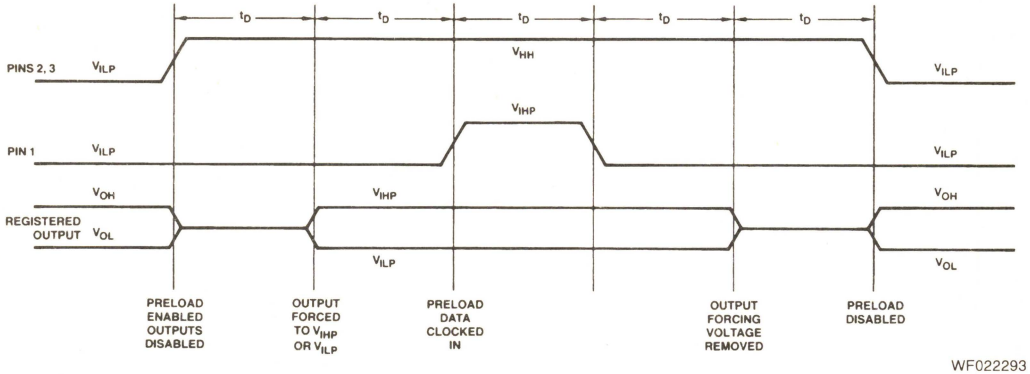

IC000900

Output Circuitry

## PRELOAD of Registered Outputs

The AmPAL22V10 registered outputs are provided with circuitry to allow loading each register synchronously with either a HIGH or LOW. This feature will simplify testing since any state can be loaded into the registers to control test sequencing.

The pin levels and timing necessary to perform the PRELOAD function are detailed below.




WF022293

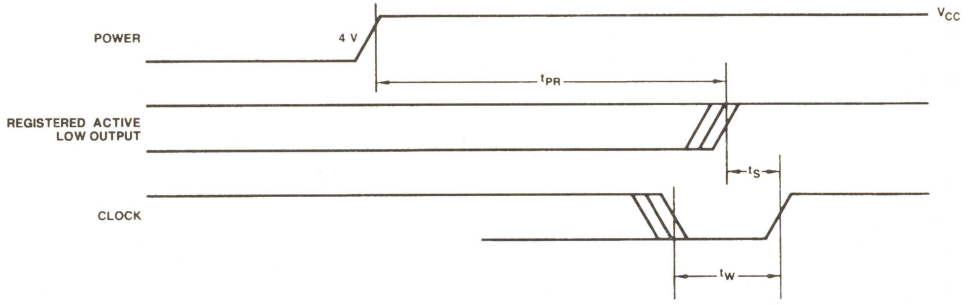
| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |

| Level forced on registered output pin during PRELOAD cycle | Register Q output state after cycle |
|---|---|
| $V_{IHP}$ | HIGH |
| $V_{ILP}$ | LOW |

## Power-Up RESET

The registered devices in the AMD PAL Family have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to LOW. The output state will depend on the polarity of the output buffer. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization. A timing diagram and parameter table are shown below. Due to the asynchronous operation of the power-up reset, and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to ensure a valid power-up reset. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from LOW to HIGH until all applicable input and feedback setup times are met.




WF022301

| Parameter Symbol | Parameter Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

# AmPAL22V10

24-Pin IMOX™ Programmable Array Logic (PAL)

## Distinctive Characteristics

- Second-generation PAL device architecture
- Increased logic power — up to 22 inputs and 10 outputs
- Increased product terms — average 12 per output
- Variable product term distribution improves ease of use
- Each output user programmable for registered or combinatorial operation
- Individually user programmable output polarity
- Extra terms provide logical synchronous PRESET and asynchronous RESET capability
- Comes in standard and high-speed versions — 18 ns typical propagation delay
- PRELOAD for improved testability
- Packaged in 24-pin Slim DIP and 28-pin chip carrier packages
- Platinum-Silicide fuses ensure high programming yield, fast programming and high reliability
- AC and DC testing done at the factory utilizing special designed-in test features

## General Description

The AmPAL22V10 is a second-generation Programmable Array Logic (PAL) device. It utilizes the familiar sum-of-products (AND-OR) logic structure, allowing users to program custom logic functions. The AmPAL22V10 is an extension of the PAL device concept. The AmPAL22V10 permits the development of custom LSI functions of 500 to 800 equivalent gate complexity.

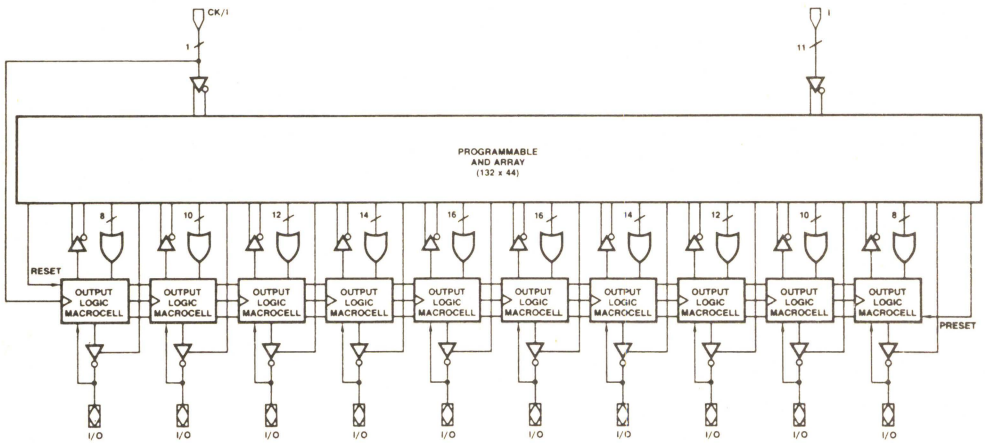
The AmPAL22V10 contains up to 22 inputs and 10 outputs. It incorporates the capability of defining and programming the architecture of each output on an individual basis. Each output is user programmable for either registered or combinatorial operation. This allows the designer to optimize the device design, by having only as many registers as needed. In addition each output has user-programmable output polarity, further simplifying design and contributing to the precise application requirements.

Increased logic power has been built into the AmPAL22V10 by increasing the number of product terms from 8-per-output to an average of 12-per-output. Further innovation can be seen in the introduction of variable product term distribution. This technique allocates from 8 to 16 logical product terms to each output (please refer to block diagram for distribution details). This variable allocation of terms allows far more complex functions to be implemented than in previous devices .

System operation has been enhanced by the addition of a synchronous-PRESET and an asynchronous-RESET product term. These terms are common to all output registers.

The AmPAL22V10 also incorporates power-up RESET and the capability to PRELOAD the output registers to any desired state during testing. PRELOAD is essential to permit full logical verification during test.

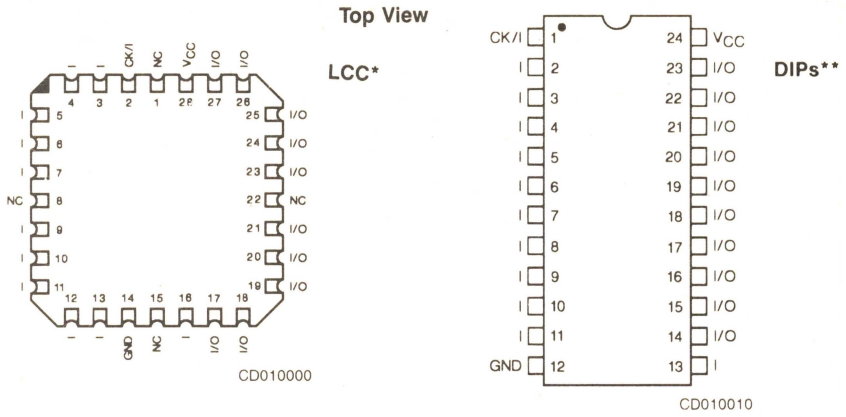
## Block Diagram




BD006590

04162F/0
JANUARY 1988

## Connection Diagram




*Also available in PLCC. Pinouts identical to LCC.
**Also available in Flatpack. Pinouts identical to DIPs.

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**

AMPAL22V10 A D C —

**E. OPTIONAL PROCESSING**
Blank = Standard processing
B = Burn-in

**D. TEMPERATURE RANGE**
C = Commercial (0 to +75°C)
E = Extended Commercial (−55 to +125°C)

**C. PACKAGE TYPE**
P = 24-Pin Plastic DIP (PD3024)
D = 24-Pin Ceramic DIP (CD3024)
J = 28-Pin Plastic Leaded Chip Carrier (PL 028)
L = 28-Pin Ceramic Leadless Chip Carrier (CL 028)

**B. SPEED OPTION**
Blank = 35 ns
A = 25 ns

**A. DEVICE NUMBER/DESCRIPTION**
AmPAL22V10
24-Pin IMOX Programmable Array Logic

| Valid Combinations | |
|---|---|
| AMPAL22V10 | PC, DC, DCB, DE, JC, LC, LE |
| AMPAL22V10A | |

#### Valid Combinations

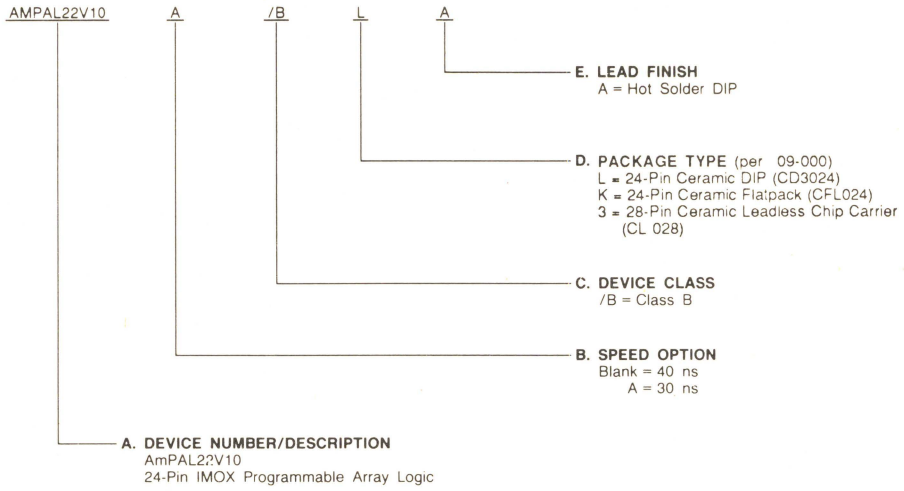
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information

### APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL22V10<br>AMPAL22V10A | /BLA/B3A/BKA |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

**Group A Tests**

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, and 11.

**DESC Certified PAL Devices**

| Generic | AMD Part Number | DESC Numbers |
|---|---|---|
| 22V10 | AmPAL22V10A/BLA | 5962-8605301LX |
| | AmPAL22V10A/B3A | 5962-86053013X |
| | AmPAL22V10A/BKA | 5962-8605301KX |

## Functional Description

The AmPAL22V10 is a second-generation Programmable Array Logic device. It contains a programmable fuse array organized in the familiar sum-of-products (AND-OR) structure.

The block diagram below shows the basic architecture of the AmPAL22V10. There up to 22 inputs and 10 outputs available. The inputs are connected to a programmable-AND array which contains 120 logical product terms. Initially the AND gates are connected, via fuses, to both the TRUE and complement of every input. By selective programming of fuses the AND gates may be "connected" to only the TRUE input (by blowing the complement fuse), to only the complement input (by blowing the TRUE fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the TRUE and complement fuses are left intact, a logical FALSE results on the output of the AND gate. An AND gate with all fuses blown will assume the logical-TRUE state. The outputs of the AND gates are connected to fixed-OR gates. There is an average of 12 product terms per OR gate (output), and as the block diagram shows, variable product term distribution has been implemented. This technique allocates different quantities of logical product terms to different outputs, allowing more complex logical functions to be performed than were previously possible. Up to 16 logical terms can be evaluated in one output in a single clock cycle (no feedback necessary).

### Output Logic Macrocells (OLMs)

A dramatic innovation in logic design is the implementation on the AmPAL22V10 of variable output architecture. This allows the user to program on an output-by-output basis the function of the outputs. As shown in the Output Logic Macrocell (OLM) diagram below, each output cell contains two additional fuses ($S_0$ and $S_1$). $S_1$ controls whether the output will be registered or combinatorial. $S_0$ controls the output polarity (active HIGH or active LOW). Depending on the states of these 2 fuses, an individual output will operate in one of four modes (see logic diagrams on next page). Registered/Active LOW; Registered/Active HIGH; Combinatorial/Active LOW; Combinatorial/Active HIGH. (Note that the feedback path also changes with output mode.) This innovation gives the designer more flexibility and enables him to optimize the device for precise application requirements. It also allows for better device utilization — you only program as many registers as are needed.

### PRESET/RESET

To improve in-system functionality, the AmPAL22V10 has additional PRESET and RESET product terms. These terms are connected to all registered outputs. When the synchronous-PRESET product term is asserted (HIGH), the output registers will be loaded with a HIGH on the next LOW-to-HIGH clock transition. When the asynchronous-RESET product term is asserted (HIGH), the output registers will be immediately loaded with a LOW (independent of the clock). These functions are particularly useful for applications such as system power-on and reset.

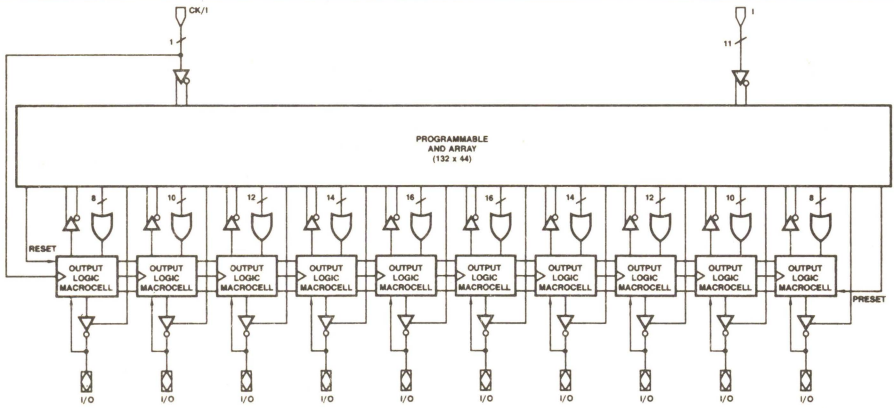
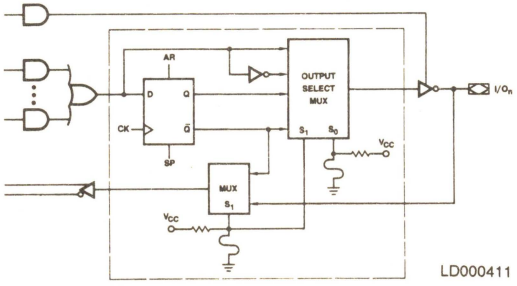



Figure 1. Block Diagram




| $S_1$ | $S_0$ | Output Configuration |
|---|---|---|
| 0 | 0 | Registered/Active LOW |
| 0 | 1 | Registered/Active HIGH |
| 1 | 0 | Combinatorial/Active LOW |
| 1 | 1 | Combinatorial/Active HIGH |

0 = Unblown Fuse
1 = Blown Fuse

Figure 2. Output Logic Macrocell Diagram

## PRELOAD

To simplify testing, the AmPAL22V10 is designed with PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the output registers.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. To verify these transitions requires the ability to set the state registers into an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is then clocked into a new state, or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

## Fabrication

The AmPAL22V10 is manufactured using Advanced Micro Devices' IMOX oxide isolation process. This advanced process permits an increase in density and a decrease in internal capacitance resulting in the fastest possible programmable logic devices.

The AmPAL22V10 is fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are preprogrammed during manufacturing to ensure extremely high field programming yields ( > 98.5%), and provide extra test paths to achieve excellent parametric correlation.

Platinum Silicide was selected as the fuse-link material to achieve a well-controlled melt rate resulting in large nonconductive gaps that ensure very stable, long-term reliability. Extensive operating testing has proven that this low-field, large-gap technology offers high reliability for fusible link programmable logic.

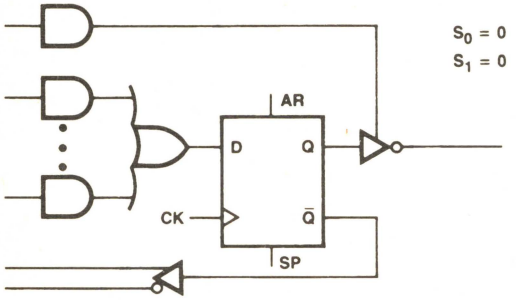



**Figure 3-1. Registered/Active LOW**

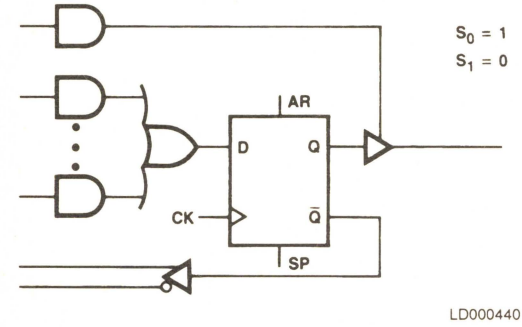



**Figure 3-2. Registered/Active HIGH**

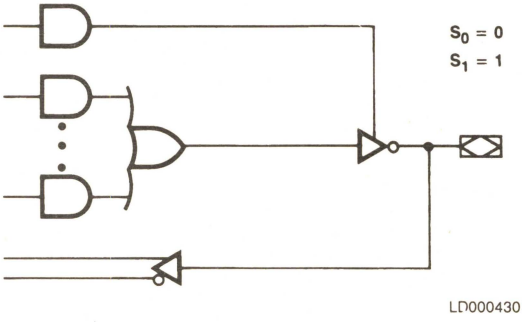



**Figure 3-3. Combinatorial/Active LOW**

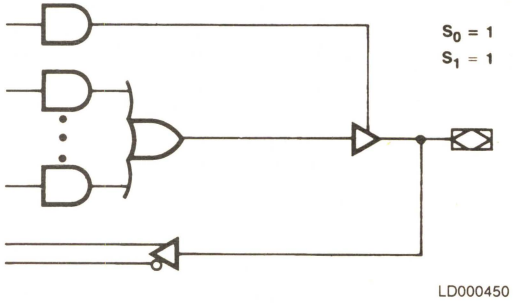
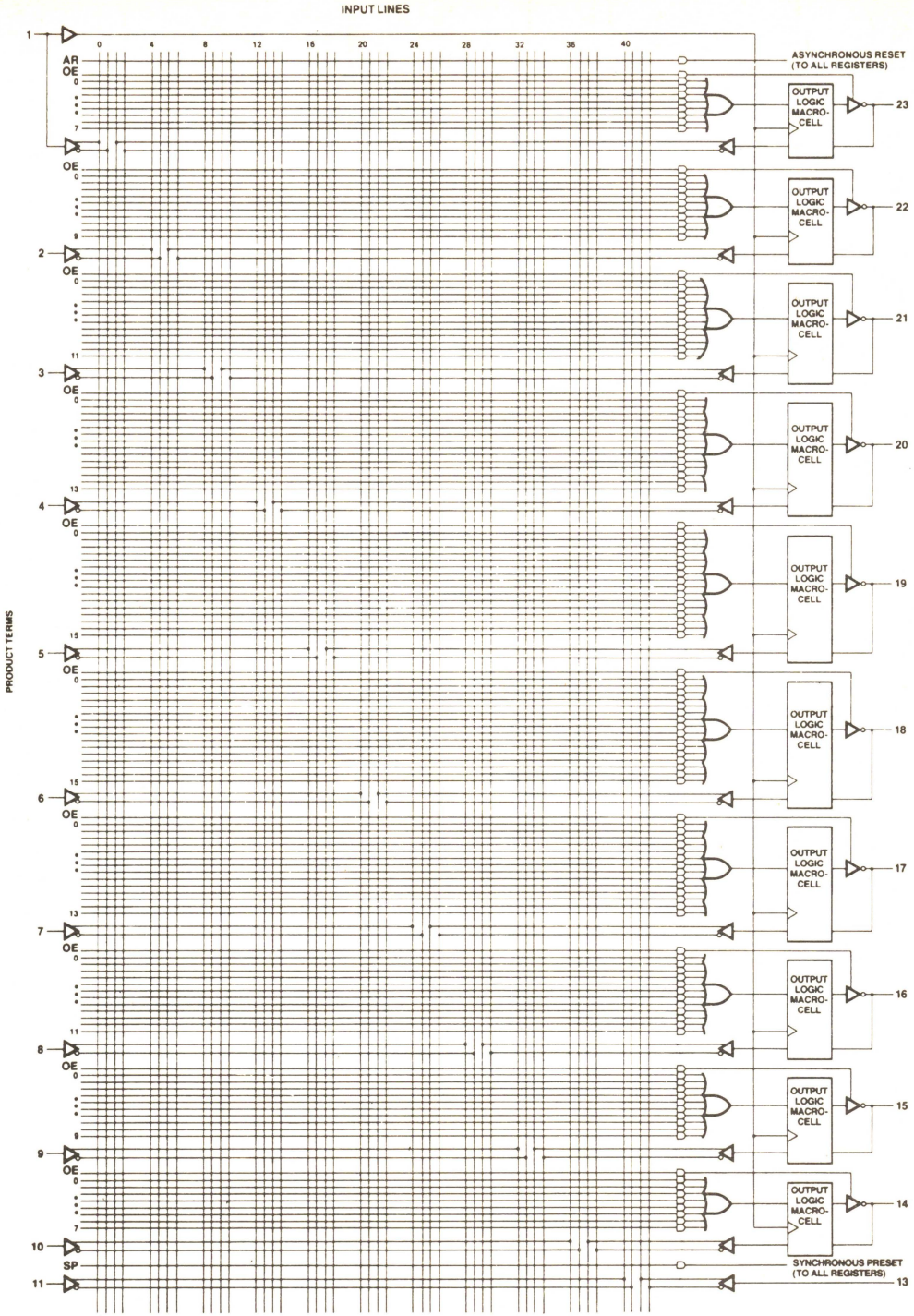



**Figure 3-4. Combinatorial/Active HIGH**




*Pinout for DIPs only.

**Figure 4. AmPAL22V10* Logic Diagram**

LD000480

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
(Pin 24 to Pin 12) Continuous ..... −0.5 to +7 V
DC Voltage Applied to Outputs
(Except During Programming) ..... −0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 16 V
Output Current into Outputs During Programming
(Max. Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5 mA
Ambient Temperature with Power Applied ..... +125°C

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) Operating Free Air ..... 0°C to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) Operating Case ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military Product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | C Devices | 2.4 | 3.5 | | Volts |
| | | | $I_{OH}$ = −2 mA | E/M Devices | | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 16 mA | C Devices | | | 0.50 | Volts |
| | | | $I_{OL}$ = 12 mA | E/M Devices | | | | |
| $V_{IH}$ (Note 2) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for all Inputs | | | 2.0 | | | Volts |
| $V_{IL}$ (Note 2) | Input LOW Level | Guaranteed Input Logical LOW Voltage for all Inputs | | | | | 0.8 | Volts |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | −20 | −100 | μA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | 25 | μA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | −30 | −50 | −90 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | | | | 150 | 180 | mA |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | −0.9 | −1.2 | Volts |
| $I_{OZH}$ | Output Leakage Current (Note 4) | Vcc = Max., $V_{IN}$ = $V_{IL}$ or $V_{IH}$ | $V_O$ = 2.7 V | | | | 100 | μA |
| $I_{OZL}$ | | | $V_O$ = 0.4 V | | | | −100 | |
| $C_{IN}$ † | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz (Note 5) | | Pins 1, 13 | | 11 | | pF |
| | | | | Others | | 6 | | |
| $C_{OUT}$ † | Output Capacitance | $V_{OUT}$ = 2.0 V @f = 1 MHz (Note 5) | | | | 9 | | pF |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).
5. Pinout for DIPs only.

† Not included in Group A tests.

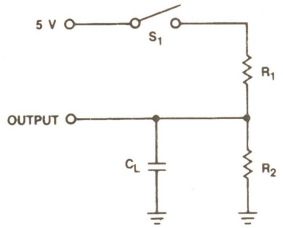
## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 7, 8, 9, 10, 11 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | Typ. (Note 1) | C Devices | | | | E/M Devices | | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | "A" | | "Std" | | "A" | | "Std" | | |
| | | | | Min. | Max. | Min. | Max. | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | Input or Feedback to Non-Registered Output | C Devices $R_1 = 300$ $R_2 = 390$ | 18 | | 25 | | 35 | | 30 | | 40 | ns |
| $t_{EA}$ | Input to Output Enable | | 18 | | 25 | | 35 | | 30 | | 40 | ns |
| $t_{ER}$ | Input to Output Disable | | 18 | | 25 | | 35 | | 30 | | 40 | ns |
| $t_{CO}$ | Clock to Output | | 10 | | 15 | | 25 | | 20 | | 25 | ns |
| $t_S$ | Input or Feedback Setup Time | | 13 | 20 | | 30 | | 25 | | 35 | | ns |
| $t_H$ | Hold Time | | −10 | 0 | | 0 | | 0 | | 0 | | ns |
| $t_P$ | Clock Period ($t_S + t_{CO}$) | | | 35 | | 55 | | 45 | | 60 | | ns |
| $t_W$ | Clock Width | | | 15 | | 25 | | 20 | | 30 | | ns |
| $f_{MAX}$ | Maximum Frequency | E/M Devices $R_1 = 390$ $R_2 = 750$ | | | 28.5 | | 18 | | 22 | | 16.5 | MHz |
| $t_{AW}$ | Asynchronous Reset Width | | | 25 | | 35 | | 30 | | 40 | | ns |
| $t_{AR}$ | Asynchronous Reset Recovery Time | | | 25 | | 35 | | 30 | | 40 | | ns |
| $t_{AP}$ | Asynchronous Reset to Registered Output Reset | | | | 30 | | 40 | | 35 | | 45 | ns |

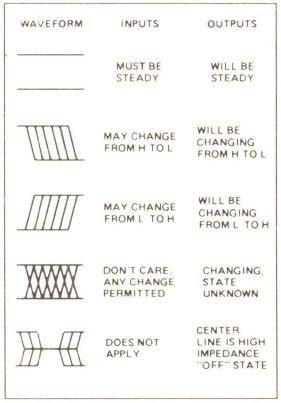
Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high-impedance to HIGH tests and closed for high-impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high-impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high-impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

## Switching Test Circuit




TC001190

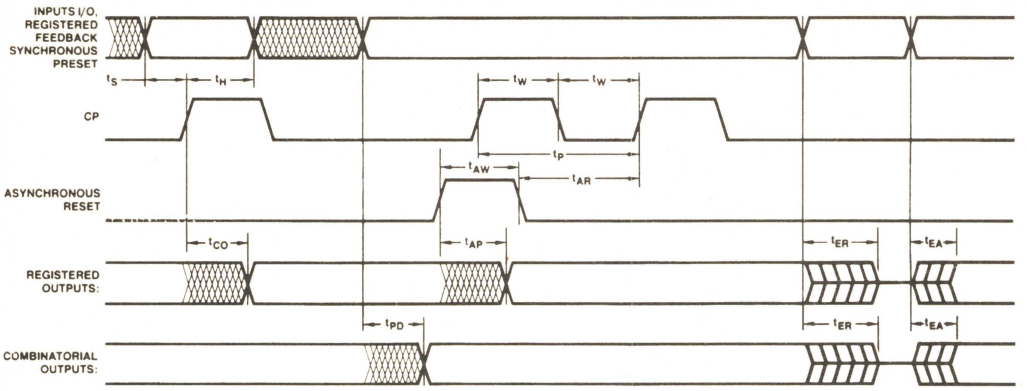
## Key to Switching Waveforms




KS000010

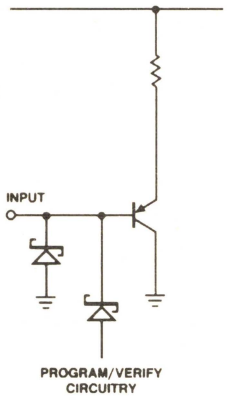
5

## Switching Waveforms



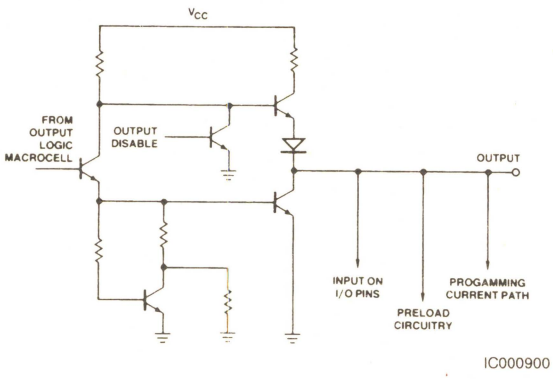

WF022280

## Input/Output Current Diagram




IC000892

Input Circuitry



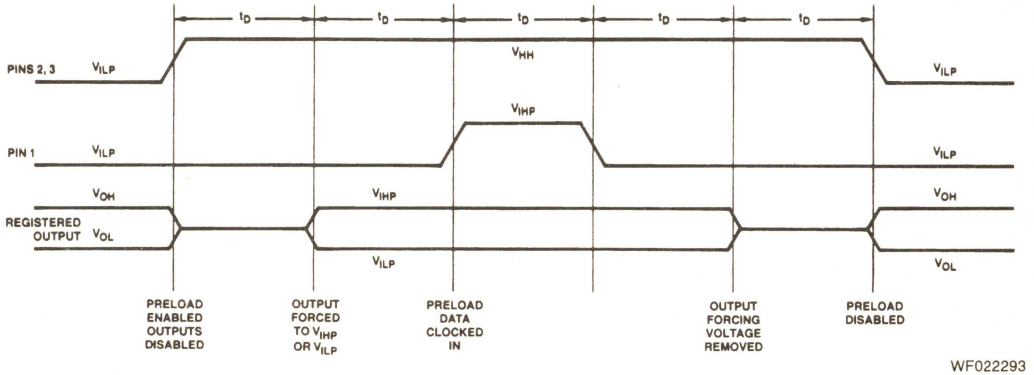

IC000900

Output Circuitry

## PRELOAD of Registered Outputs

The AmPAL22V10 registered outputs are provided with circuitry to allow loading each register synchronously with either a HIGH or LOW. This feature will simplify testing since any state can be loaded into the registers to control test sequencing.

The pin levels and timing necessary to perform the PRELOAD function are detailed below. Parameters are listed in the Programming Parameters table (page 12).




WF022293

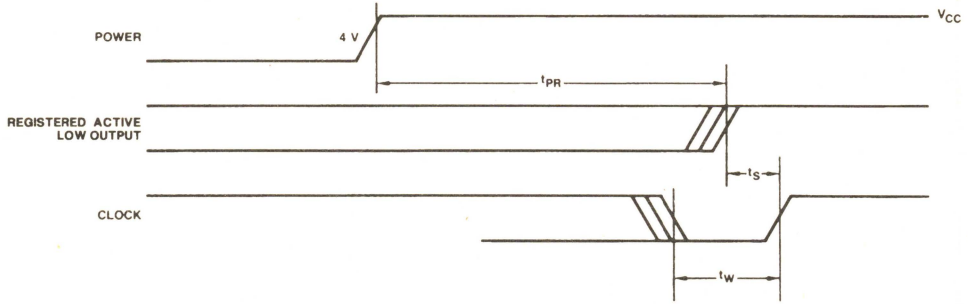
| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |

| Level forced on registered output pin during PRELOAD cycle | Register Q output state after cycle |
|---|---|
| $V_{IHP}$ | HIGH |
| $V_{ILP}$ | LOW |

### Power-up RESET

The registered devices in the AMD PAL Family have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to LOW. The output state will depend on the polarity of the output buffer. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization. A timing diagram and parameter table are shown below. Due to the asynchronous operation of the power-up reset and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to insure a valid power-up reset. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from LOW to HIGH until all applicable input and feedback setup times are met.




WF022301

| Parameter Symbol | Parameter Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

### Security Fuse Programming

A single fuse is provided on each AmPAL22V10 part to prevent unauthorized copying of PAL fuse patterns. Once blown, the circuitry enabling fuse verification and registered output PRELOAD is permanently disabled.

Programming of the security fuse is the same as an array fuse. Verification of a blown security fuse is accomplished by verifying the whole fuse array as if every fuse is blown.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

# 24-Pin XOR AmPAL20XRP10 Family

24-Pin IMOX™ Programmable Array Logic (PAL) Elements

## Distinctive Characteristics

- AND-OR-XOR logic structure
- AMD's superior IMOX technology
  - Guarantees $t_{PD}$ = 20 ns max
- Individually programmable output polarity on each output
- Eight logical product terms per output
- Programming yields > 98% are realized via platinum-silicide fuse technology and the use of added test words
- Post Programming Functional Yield (PPFY) of 99.9%
- PRELOAD feature permits full logical verification
- Reliability assured through more than 70 billion fuse hours of life testing with no failures
- AC and DC parametric testing at the factory through on-board testing circuitry
- > 3000V ESD protection per pin
- JEDEC-Standard LCC and PLCC pinout

## General Description

AMD 24-pin XOR PAL devices are high-speed, electrically programmable array logic elements. They utilize the familiar sum-of-products (AND-OR-XOR) structure allowing users to program custom logic functions to fit most applications precisely. Typically they are a replacement for low-power Schottky SSI/MSI logic circuits that require an exclusive-OR function, reducing chip count by more than 5 to 1 and greatly simplifying prototyping and board layout.

Five different devices are available, including both registered and combinatorial devices. All devices have user-programmable output polarity on all outputs. A variety of speed options allow the designer maximum flexibility in matching precise system requirements. The Product Selector Guide below shows the available speed options. The second table gives details about the functionality of the five available devices.

Please see the following pages for Block Diagrams.

## Product Selector Guide

**AMD PAL Speed/Power Families**

| Family | $t_{PD}$ ns (Max.) | | $t_S$ ns (Min.) | | $t_{CO}$ ns (Max.) | | $I_{CC}$ mA (Max.) | $I_{OL}$ mA (Min.) | |
|---|---|---|---|---|---|---|---|---|---|
| | C Devices | M Devices | C Devices | M Devices | C Devices | M Devices | C/M Devices | C Devices | M Devices |
| Very High-Speed (-20 & -25) Versions | 20 | 25 | 20 | 25 | 13 | 15 | 210 | 24 | 12 |
| High-Speed (-30 & -35) Versions | 30 | 35 | 30 | 35 | 15 | 25 | 180 | 24 | 12 |
| High-Speed, Half-Power (-30L & -35L) Versions | 30 | 35 | 30 | 35 | 15 | 25 | 105 | 24 | 12 |
| Half-Power (-40L & -45L) Versions | 40 | 45 | 40 | 45 | 30 | 35 | 105 | 24 | 12 |

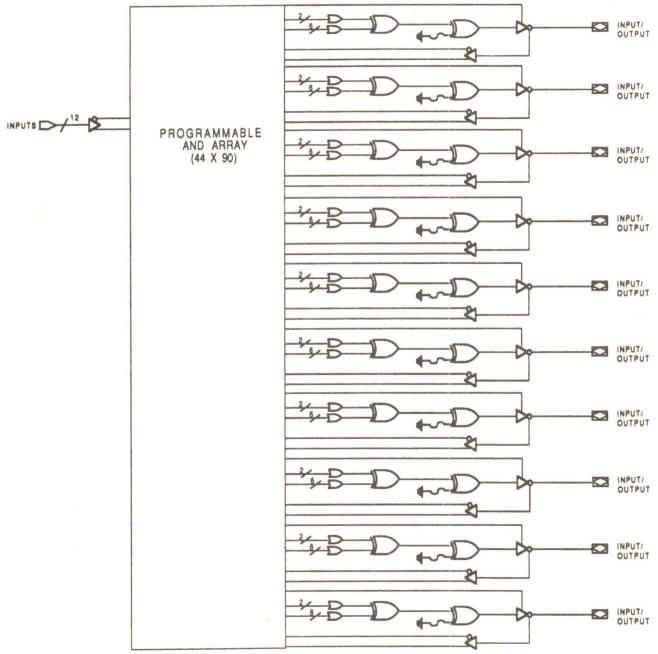
| Part Number | Array Inputs | Logic | Output Enable | Outputs/Polarity | Package Pins |
|---|---|---|---|---|---|
| 22XP10 | 12 Dedicated, 10 Bidirectional | Ten (2-6)-Wide AND-OR-XOR | Programmable | Bidirectional/Programmable | 24 |
| 20XRP4 | 10 Dedicated, 4 Feedback, 6 Bidirectional | Four (2-6)-Wide AND-OR-XOR | Dedicated | Registered/Programmable | 24 |
| | | Six 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20XRP6 | 10 Dedicated, 6 Feedback, 4 Bidirectional | Six (2-6)-Wide AND-OR-XOR | Dedicated | Registered/Programmable | 24 |
| | | Four 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20XRP8 | 10 Dedicated, 8 Feedback, 2 Bidirectional | Eight (2-6)-Wide AND-OR-XOR | Dedicated | Registered/Programmable | 24 |
| | | Two 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20XRP10 | 10 Dedicated, 10 Feedback | Ten (2-6)-Wide AND-OR-XOR | Dedicated | Registered/Programmable | 24 |



08655C/0
JANUARY 1988

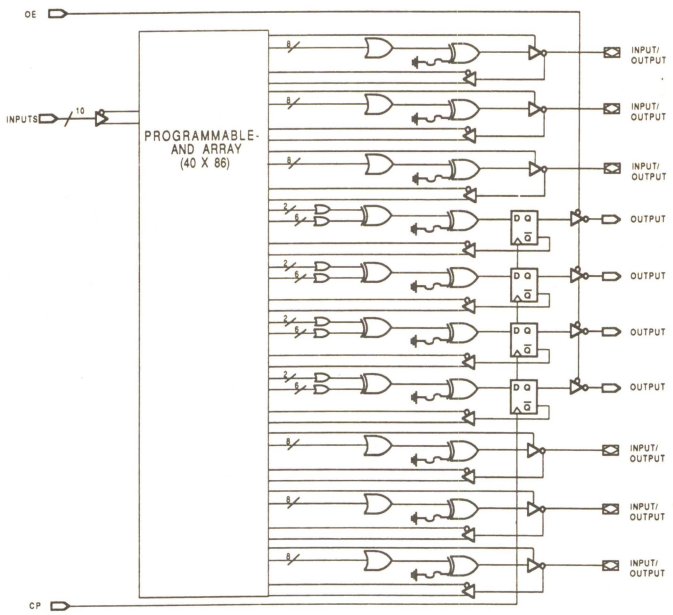
## Block Diagrams

### AmPAL22XP10




LD000900

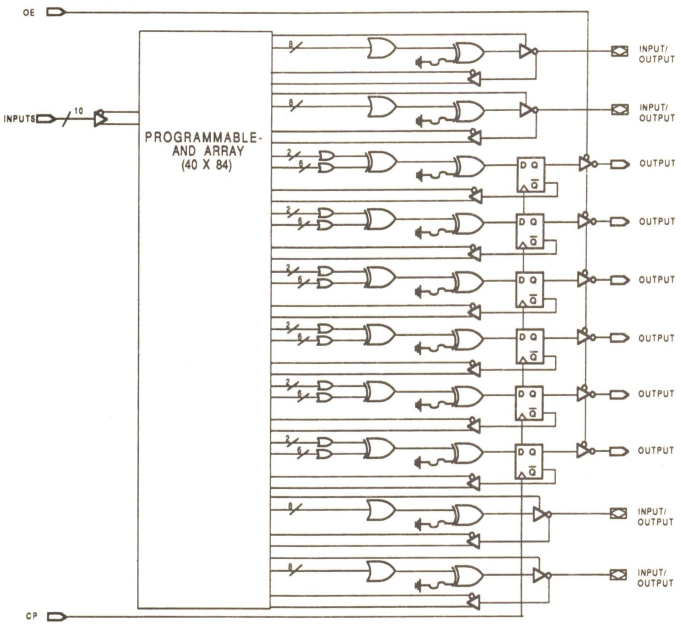
### AmPAL20XRP4




LD000940

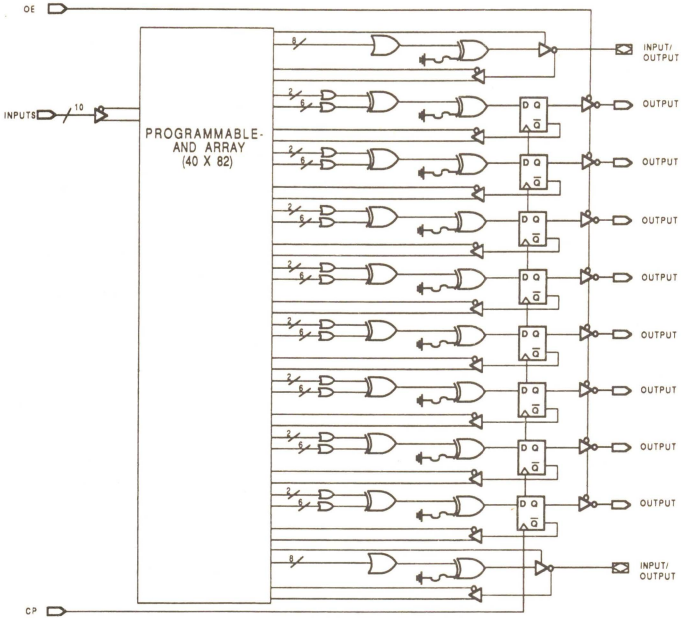
## Block Diagrams (Cont'd.)

### AmPAL20XRP6




LD000920

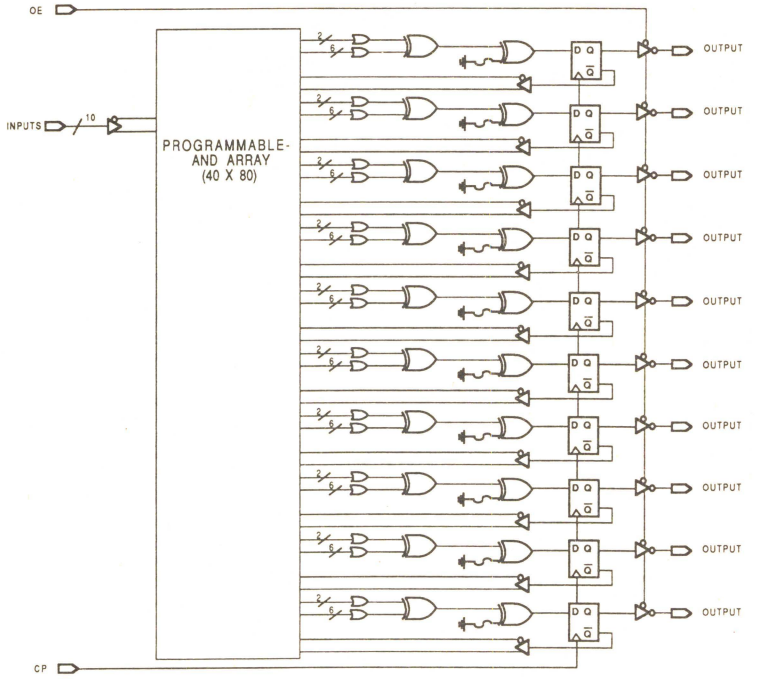
### AmPAL20XRP8




LD000930

## Block Diagrams (Cont'd.)

### AmPAL20XRP10

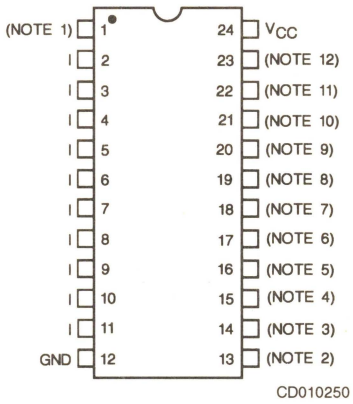



LD000910

## Connection Diagrams

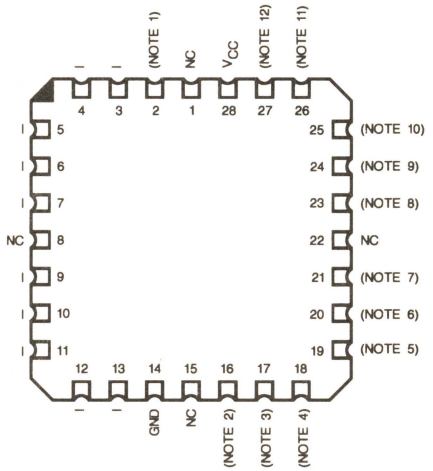
Top View

DIPs*




CD010250

LCC**




CD010260

Note: Pin 1 is marked for orientation.

Notes:

| | 22XP10 | 20XRP4 | 20XRP6 | 20XRP8 | 20XRP10 |
|---|---|---|---|---|---|
| 1 | I | CLK | CLK | CLK | CLK |
| 2 | I | OE | OE | OE | OE |
| 3 | I/O | I/O | I/O | I/O | O |
| 4 | I/O | I/O | I/O | O | O |
| 5 | I/O | I/O | O | O | O |
| 6 | I/O | O | O | O | O |
| 7 | I/O | O | O | O | O |
| 8 | I/O | O | O | O | O |
| 9 | I/O | O | O | O | O |
| 10 | I/O | I/O | O | O | O |
| 11 | I/O | I/O | I/O | O | O |
| 12 | I/O | I/O | I/O | I/O | O |

*Also available in 24-Pin Ceramic Flatpack. Pinouts identical to DIPs.
**Also available in 28-Pin Plastic Leaded Chip Carrier. Pinouts identical to LCC.

## Pin Designations

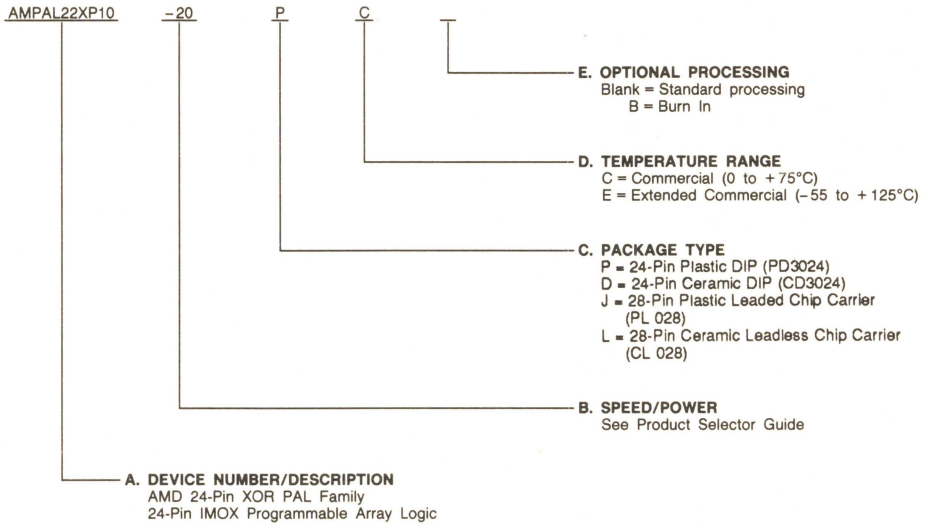
I = Input
I/O = Input/Output
O = Output
$V_{CC}$ = Supply Voltage
GND = Ground
CLK = Clock
OE = Output Enable
NC = No Connect

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AMPAL22XP10-20/-30/-30L/-40L | PC, DC, DCB, DE, JC, LC, LE |
| AMPAL20XRP4-20/-30/-30L/-40L | |
| AMPAL20XRP6-20/-30/-30L/-40L | |
| AMPAL20XRP8-20/-30/-30L/-40L | |
| AMPAL20XRP10-20/-30/-30L/-40L | |

**Valid Combinations**

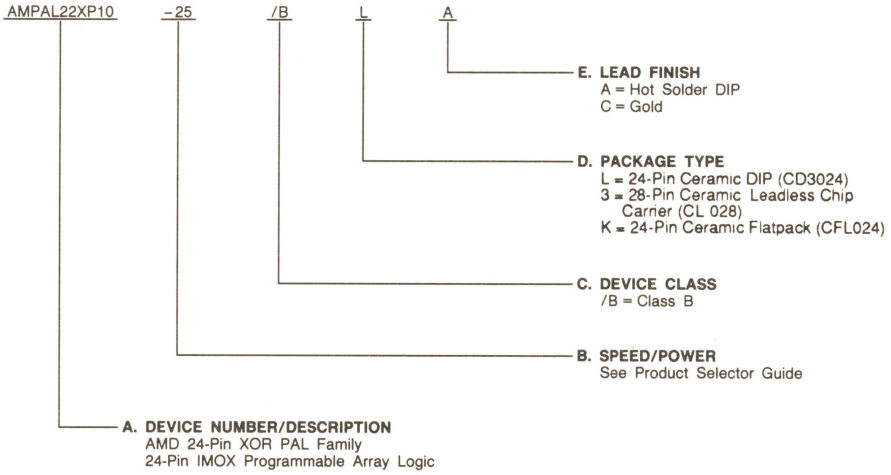
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information (Cont'd.)

### APL Products

AMD products for Aereospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL22XP10-25/-35/-35L/-45L | /BLA, /B3A, /BKA |
| AMPAL20XRP4-25/-35/-35L/-45L | |
| AMPAL20XRP6-25/-35/-35L/-45L | |
| AMPAL20XRP8-25/-35/-35L/-45L | |
| AMPAL20XRP10-25/-35/-35L/-45L | |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

**Group A Tests**

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, & 11

5

## Functional Description

### AMD 24-PIN XOR PAL20XRP10 Family Characteristics

All members of the AMD 24-Pin XOR PAL Family have common electrical characteristics and programming procedures. All parts are produced with a fusible link at each input to the AND gate array, and connections may be selectively removed by applying appropriate voltages to the circuit.

Initially the AND gates are connected, via fuses, to both the true and complement of each input. By selective programming of fuses the AND gates may be "connected" to only the true input (by blowing the complement fuse), to only the complement input (by blowing the true fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the true and complement fuses are left intact a logical false results on the output of the AND gate, while all fuses blown results in a logical true state. On the AmPAL22XP10 device, the AND gates are connected to fixed (2-6) OR-XOR structures whose outputs become device outputs. The remaining four (registered) devices function as follows: for combinatorial outputs, the AND gates are connected to fixed-OR gates whose outputs become device outputs. For registered outputs, the AND gates are connected to fixed (2-6) OR-XOR structures whose outputs become output register inputs.

All parts are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are pre-programmed during manufacturing to insure extremely high field programming yields ( > 98%), and provide extra test paths to achieve excellent parametric correlation.

### Power-Up Reset

The registered devices in the AMD PAL family have been designed to reset during system power-up. Following power-up, all registers will be initialized to zero, setting all the outputs to a logic 1. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

### PRELOAD

AMD PAL devices are designed with unique PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the registered output of an AMD PAL device.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

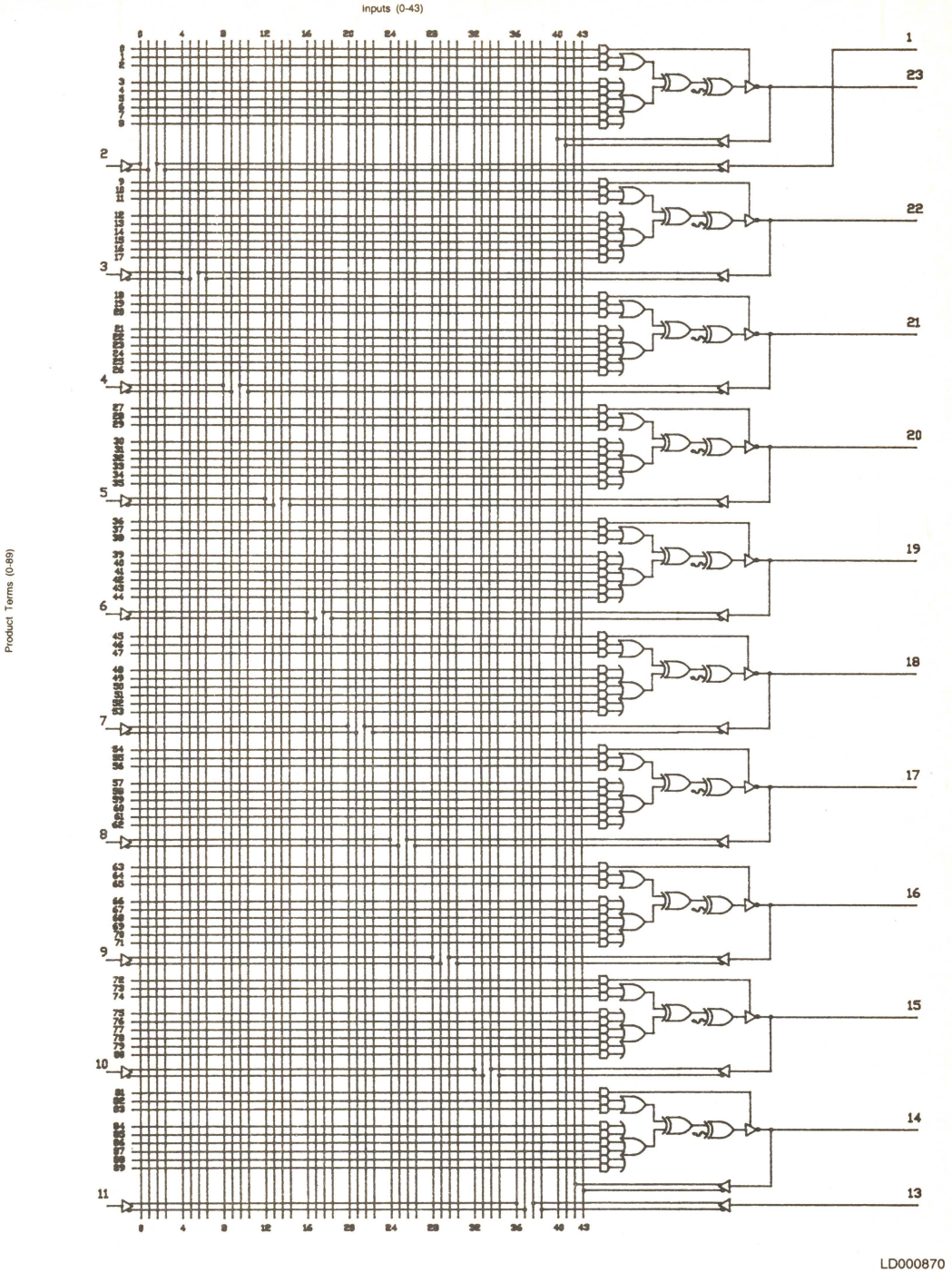
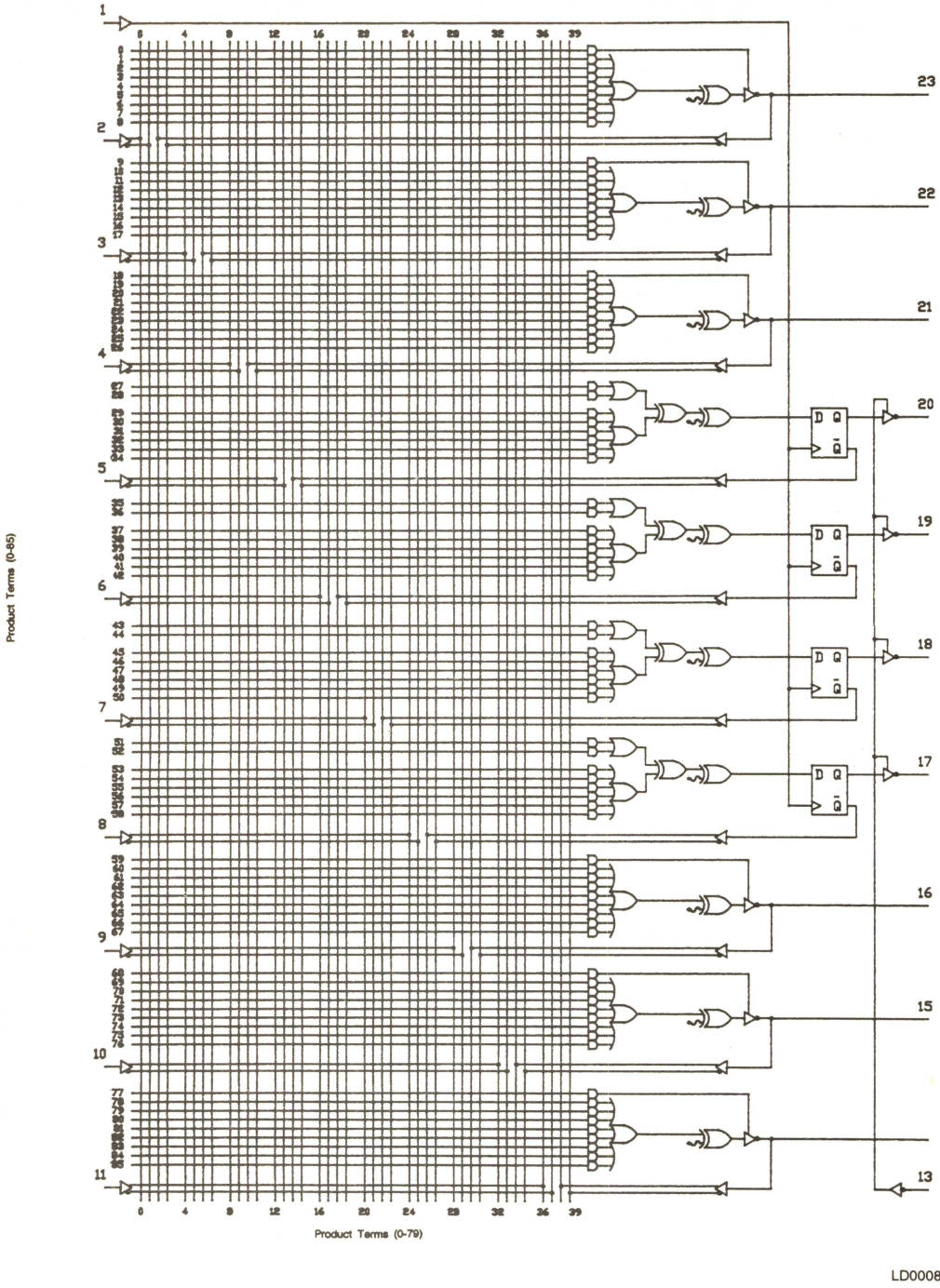



Figure 1. AmPAL22XP10 Logic Diagram




LD000860

Figure 2. AmPAL20XRP4 Logic Diagram

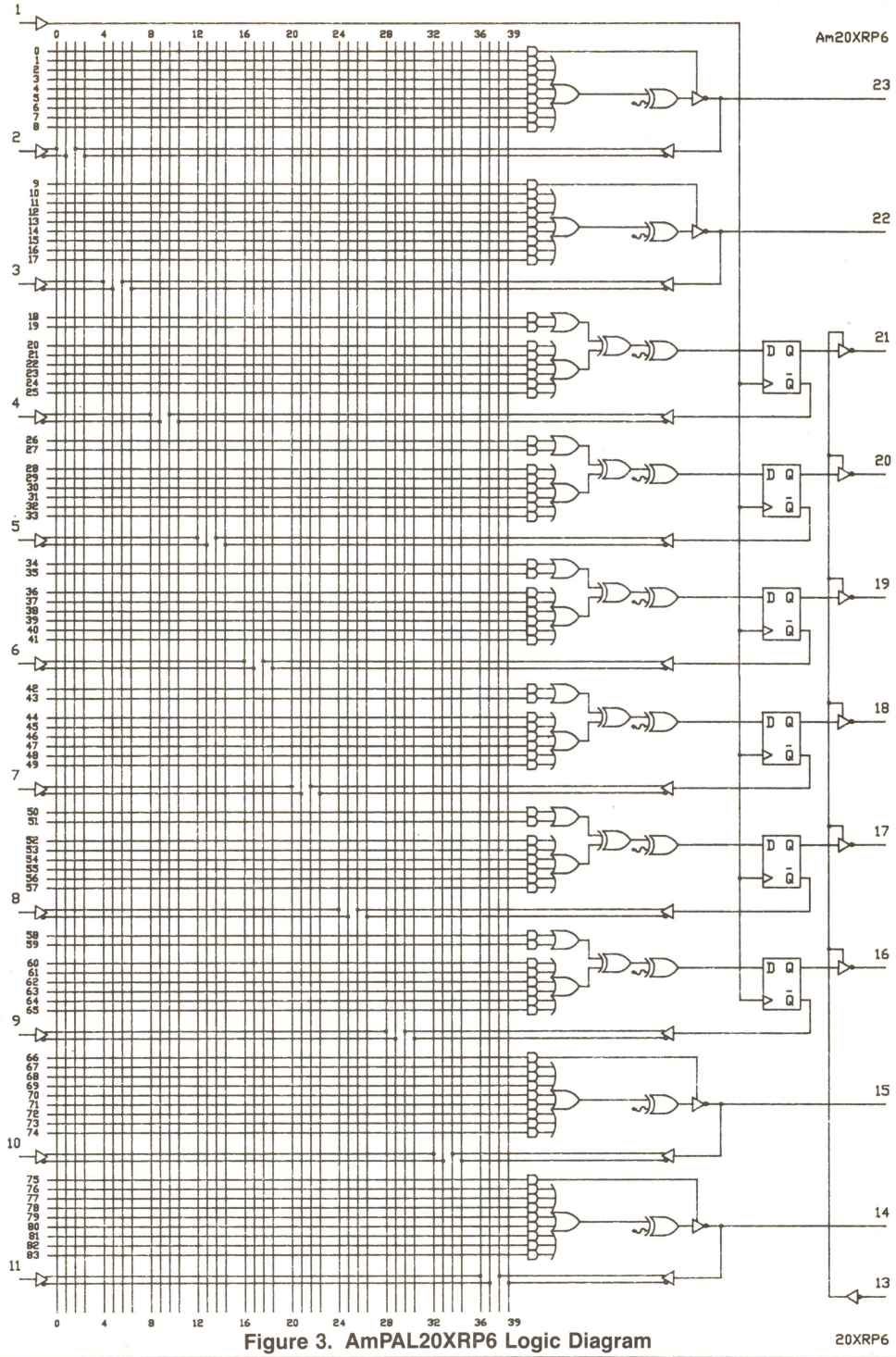
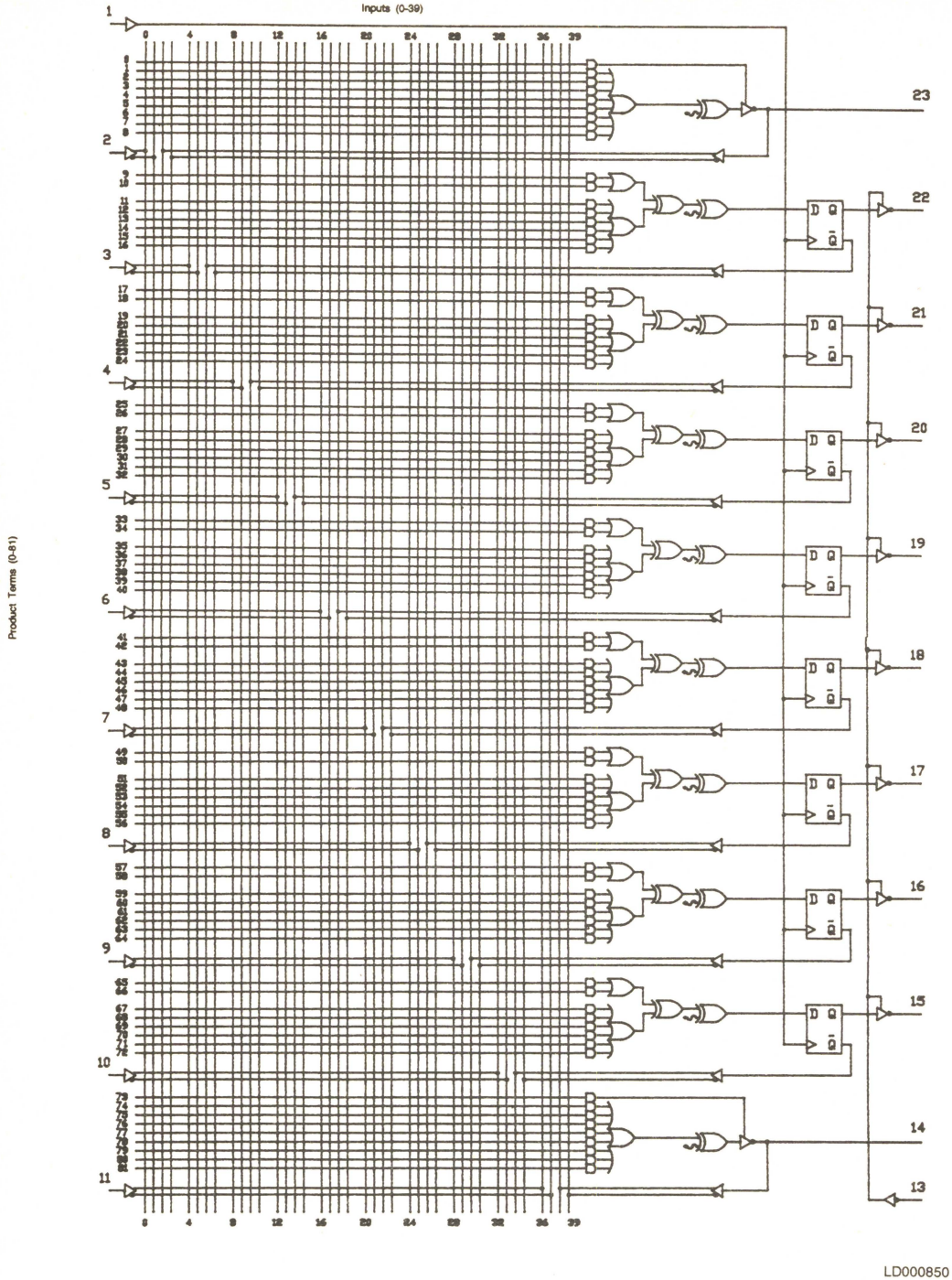



Figure 3. AmPAL20XRP6 Logic Diagram




LD000850

Figure 4. AmPAL20XRP8 Logic Diagram

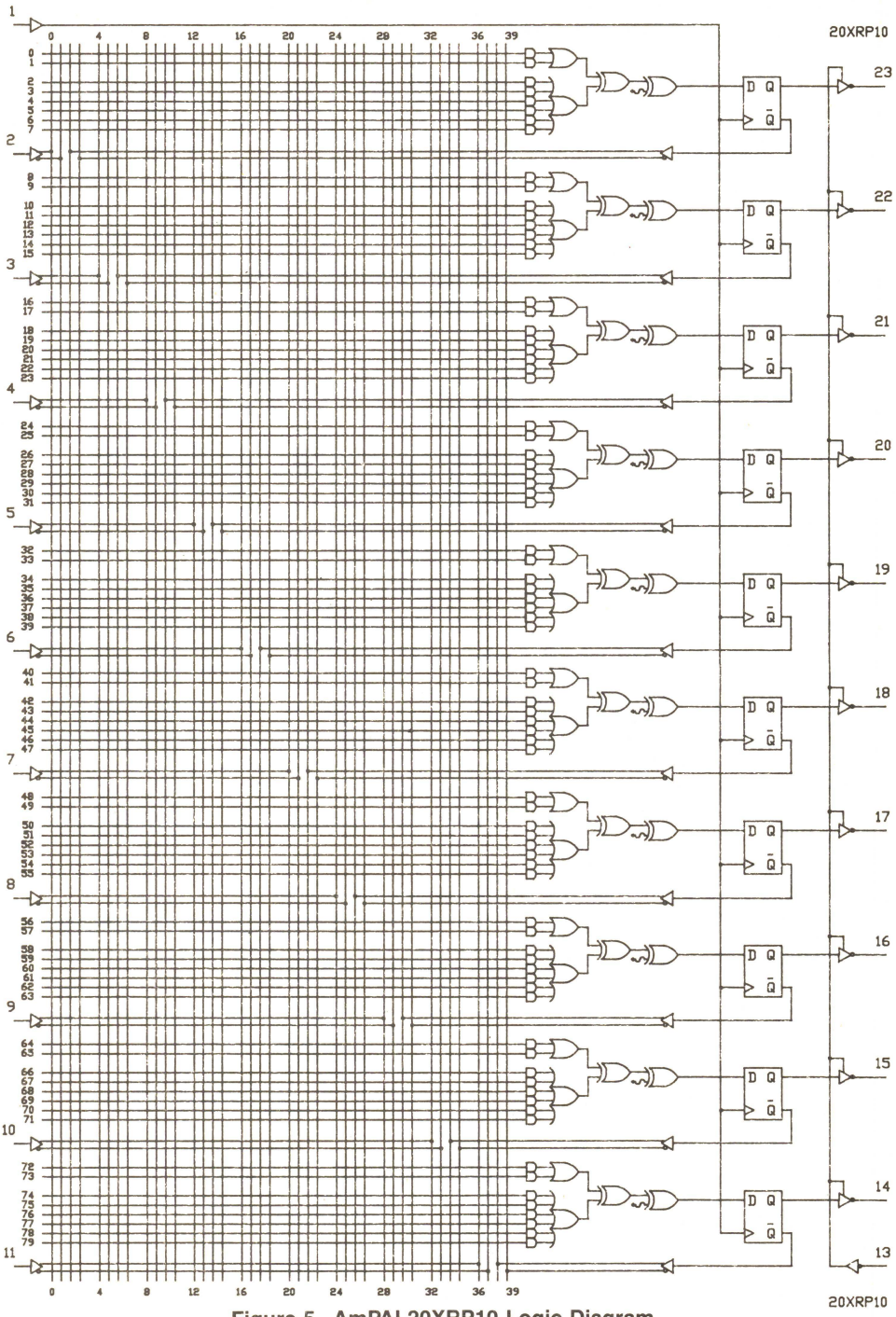
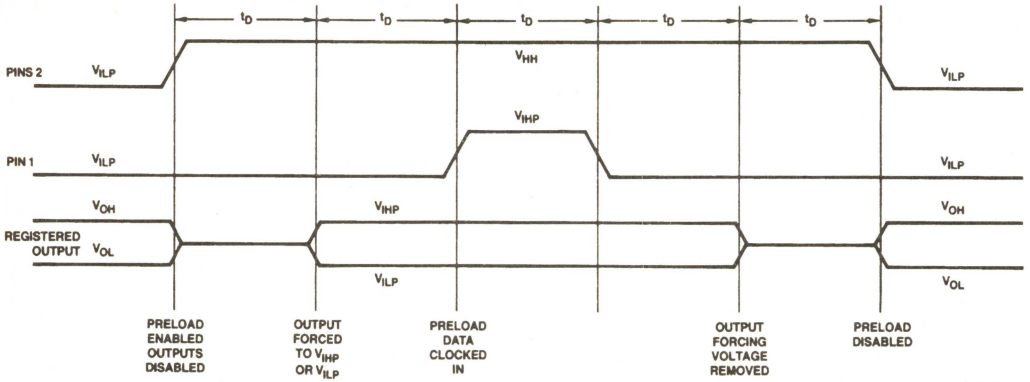



Figure 5. AmPAL20XRP10 Logic Diagram

## PRELOAD of Registered Outputs

The AMD 24-pin XOR PAL devices incorporate circuitry to allow loading each register synchronously to either a HIGH or LOW state. This feature simplifies testing since any initial state for the registers can be set to optimize test sequencing.

The pin levels and timing necessary to perform the PRELOAD function are detailed below:




WF022294

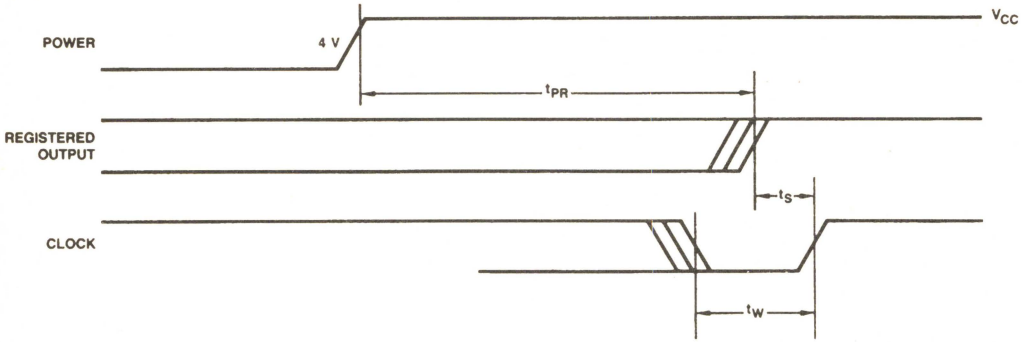
| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |

| Level forced on registered output pin during PRELOAD cycle | Register Q output state after cycle |
|---|---|
| $V_{IHP}$ | HIGH |
| $V_{ILP}$ | LOW |

## Power-Up Reset

The registered devices in the AMD 24-Pin XOR PAL Family have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to LOW. The output state will be HIGH. This feature provides flexibility to the designer and is especially valuable in simplifying state-machine initialization. A timing diagram and parameter table are shown below. Due to the asynchronous operation of the power-up RESET and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to ensure a valid power-up RESET. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from LOW to HIGH until all applicable input and feedback setup times are met.




WF022300

| Parameters | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
(Pin 24 to Pin 12) Continuous ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs
(Except During Programming).....−0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 16 V
Output Current Into Outputs During
Programming (Max Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) ..... 0 to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | COM'L | 2.4 | 3.5 | | V |
| | | | $I_{OH}$ = −2 mA | MIL | | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 24 mA | COM'L | | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | MIL | | | | |
| $V_{IH}$ (Note 2) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | 2.0 | | 5.5 | V |
| $V_{IL}$ (Note 2) | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs | | | | | 0.8 | V |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | −20 | −100 | µA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | 25 | µA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | −30 | −60 | −90 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | COM'L | MIL. | | | | mA |
| | | | −20 | −25 | | | 210 | |
| | | | −30, | −35, | | | 180 | |
| | | | −30L, −40L | −35L, −45L | | | 105 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | −0.9 | −1.2 | V |
| $I_{OZH}$ | Output Leakage Current (Note 4) | $V_{CC}$ = Max., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $V_O$ = 2.7 V | | | | 100 | µA |
| $I_{OZL}$ | | | $V_O$ = 0.4 V | | | | −100 | |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).

### Capacitance*

| Parameter Symbol | Parameter Description | Test Conditions | | Typ. | Units |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz | Pins 1, 13 | 11 | pF |
| | | | Others | 6 | |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | | 9 | |

*These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

### Commercial Range

| No. | Parameter Symbol | Parameter Description | -20 Version | | | -30 & -30L Version | | | -40L Version | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 20 | | | 30 | | | 40 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 20 | | | 30 | | | 40 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 20 | | | 30 | | | 40 | ns |
| 4 | $t_{PZX}$ | Pin 13 to Output Enable 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 15 | | | 20 | | | 35 | ns |
| 5 | $t_{PXZ}$ | Pin 13 to Output Disable 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 15 | | | 20 | | | 35 | ns |
| 6 | $t_{CO}$ | Clock to Output 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 13 | | | 15 | | | 30 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 20XRP4, 20XRP6, 20XRP8, 20XRP10 | 20 | | | 30 | | | 40 | | | ns |
| 8 | $t_H$ | Hold Time 20XRP4, 20XRP6, 20XRP8, 20XRP10 | 0 | | | 0 | | | 0 | | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 33 | | | 45 | | | 70 | | | ns |
| 10 | $t_W$ | Clock Width | 10 | | | 15 | | | 25 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 30.3 | | | 22.2 | | | 14.3 | MHz |

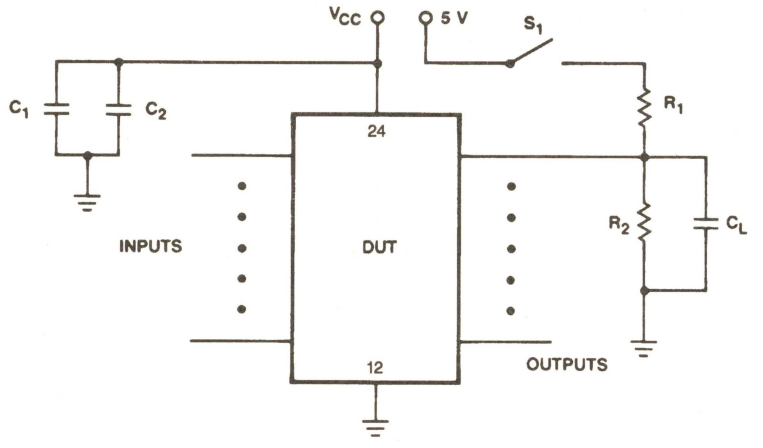
Notes:
1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

### Military Range

| No. | Parameter Symbol | Parameter Description | -25 Version | | | -35 & -35L Version | | | -45L Version | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 25 | | | 35 | | | 45 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 25 | | | 35 | | | 45 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 22XP10, 20XRP4, 20XRP6, 20XRP8 | | | 25 | | | 35 | | | 45 | ns |
| 4 | $t_{PZX}$ | Pin 13 to Output Enable 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 20 | | | 25 | | | 40 | ns |
| 5 | $t_{PXZ}$ | Pin 13 to Output Disable 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 20 | | | 25 | | | 40 | ns |
| 6 | $t_{CO}$ | Clock to Output 20XRP4, 20XRP6, 20XRP8, 20XRP10 | | | 15 | | | 25 | | | 35 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 20XRP4, 20XRP6, 20XRP8, 20XRP10 | 25 | | | 35 | | | 45 | | | ns |
| 8 | $t_H$ | Hold Time 20XRP4, 20XRP6, 20XRP8, 20XRP10 | 0 | | | 0 | | | 0 | | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 40 | | | 60 | | | 80 | | | ns |
| 10 | $t_W$ | Clock Width | 12 | | | 20 | | | 30 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 25 | | | 16.7 | | | 12.5 | MHz |

Notes:
1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

## Switching Test Circuit

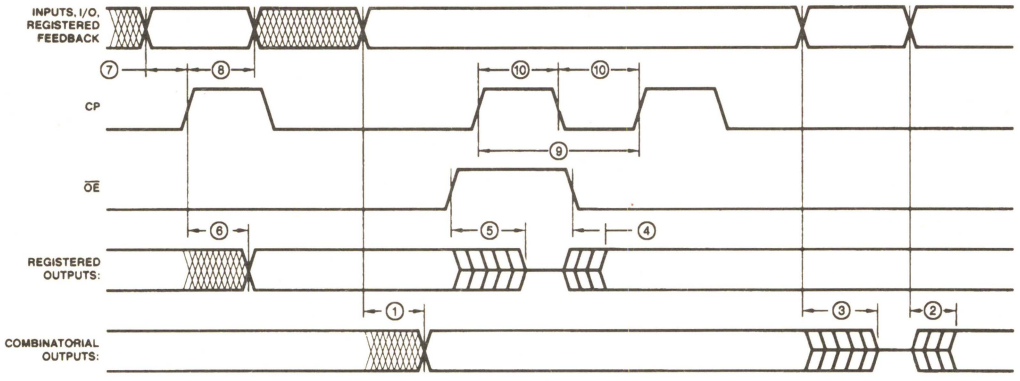



Note: $C_1$ and $C_2$ are to bypass $V_{CC}$ to ground.

| TEST OUTPUT LOADS | | |
|---|---|---|
| **Pin Name** | **Commercial** | **Military** |
| $R_1$ | 200 Ω | 390 Ω |
| $R_2$ | 390 Ω | 750 Ω |
| $C_1$ | 1 μF | 1 μF |
| $C_2$ | 0.1 μF | 0.1 μF |
| $C_L$ | 50 pF | 50 pF |

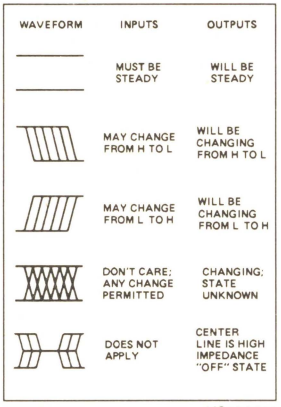
5

## Switching Waveforms



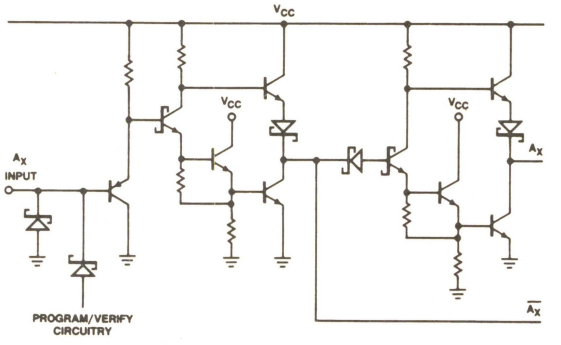

WF002571

## Key to Timing Diagram



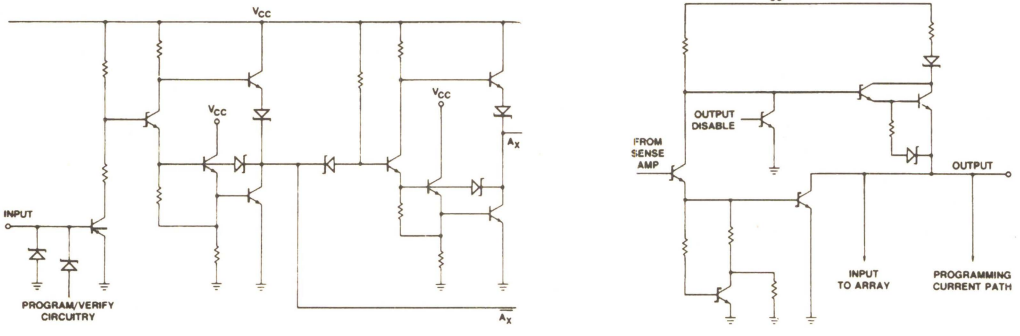

KS000010

## Input Circuitry




IC000720

## Output Circuitry




IC000801

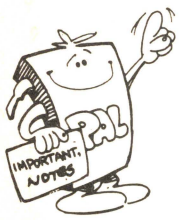
### Security Fuse Programming

A single fuse is provided on each device to prevent unauthorized copying of PAL fuse patterns. Once blown, the circuitry enabling fuse verification and registered output PRELOAD is permanently disabled.

Programming of the security fuse is the same as an array fuse. Verification of a blown security fuse is accomplished by verifying the whole fuse array as if every fuse is blown.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)


IMPORTANT
NOTES

# 24-Pin Enhanced AmPAL20RP10 Family

24-Pin IMOX™ Programmable Array Logic (PAL) Elements

## Distinctive Characteristics

- AMD's superior IMOX technology
  - Guarantees $t_{PD}$ = 15 ns max
- Individually programmable output polarity on each output
- Eight logical product terms per output
- Programming yields > 98% are realized via platinum-silicide fuse technology and the use of added test words
- Post Programming Functional Yield (PPFY) of 99.9%
- PRELOAD feature permits full logical verification
- Reliability assured through more than 70 billion fuse hours of life testing with no failures
- AC and DC parametric testing at the factory through on-board testing circuitry
- > 3000V ESD protection per pin
- JEDEC-Standard LCC and PLCC pinout

## General Description

AMD Enhanced 24-pin PAL devices are high-speed, electrically programmable array logic elements. They utilize the familiar sum-of-products (AND-OR) structure allowing users to program custom logic functions to fit most applications precisely. Typically they are a replacement for low-power Schottky SSI/MSI logic circuits, reducing chip count by more than 5 to 1 and greatly simplifying prototyping and board layout. Additional product terms, two additional outputs, and programmable output polarity are enhancements over industry-standard 24-pin PAL devices.

Five different devices are available, including both registered and combinatorial devices. All devices have user-programmable output polarity on all outputs. A variety of speed options allow the designer maximum flexibility in matching precise system requirements. The Product Selector Guide below shows the available speed options. The second table gives details about the functionality of the five available devices.

Please see the following pages for Block Diagrams.

## Product Selector Guide

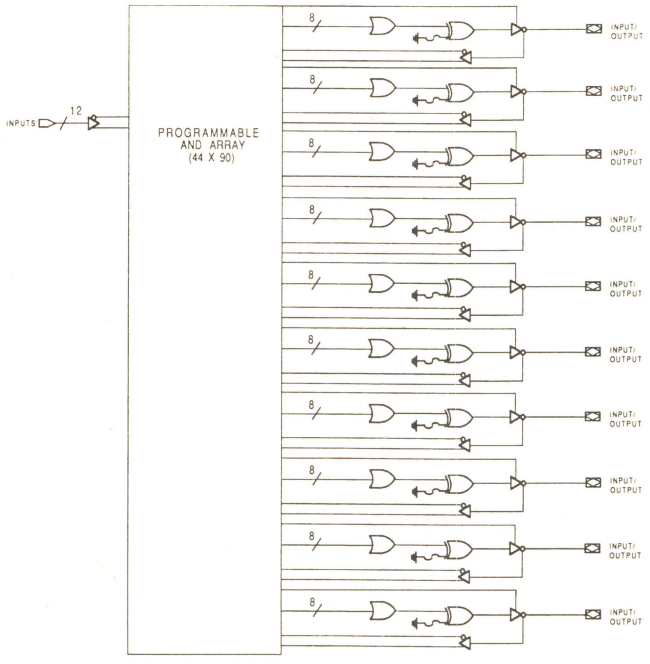
**AMD PAL Speed/Power Families**

| Family | $t_{PD}$ ns (Max.) | | $t_S$ ns (Min.) | | $t_{CO}$ ns (Max.) | | $I_{CC}$ mA (Max.) | $I_{OL}$ mA (Min.) | |
|---|---|---|---|---|---|---|---|---|---|
| | C Devices | M Devices | C Devices | M Devices | C Devices | M Devices | C/M Devices | C Devices | M Devices |
| Very High-Speed ("B") Versions | 15 | 20 | 15 | 20 | 12 | 15 | 210 | 24 | 12 |
| High-Speed ("A") Versions | 25 | 30 | 25 | 30 | 15 | 20 | 210 | 24 | 12 |
| High-Speed, Half-Power ("AL") Versions | 25 | 30 | 25 | 30 | 15 | 20 | 105 | 24 | 12 |
| -20 & -25 Versions (AmPAL20L10 only) | 20 | 25 | N/A | N/A | N/A | N/A | 165 | 24 | 12 |

| Part Number | Array Inputs | Logic | Output Enable | Outputs/Polarity | Package Pins |
|---|---|---|---|---|---|
| 22P10 | 12 Dedicated, 10 Bidirectional | Ten (8)-Wide AND-OR | Programmable | Bidirectional/Programmable | 24 |
| 20RP4 | 10 Dedicated, 4 Feedback, 6 Bidirectional | Four (8)-Wide AND-OR | Dedicated | Registered/Programmable | 24 |
| | | Six 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20RP6 | 10 Dedicated, 6 Feedback, 4 Bidirectional | Six (8)-Wide AND-OR | Dedicated | Registered/Programmable | 24 |
| | | Four 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20RP8 | 10 Dedicated, 8 Feedback, 2 Bidirectional | Eight (8)-Wide AND-OR | Dedicated | Registered/Programmable | 24 |
| | | Two 8-Wide AND-OR | Programmable | Bidirectional/Programmable | |
| 20RP10 | 10 Dedicated, 10 Feedback | Ten (8)-Wide AND-OR | Dedicated | Registered/Programmable | 24 |
| 20L10 | 12 Dedicated, 8 Bidirectional | Ten (3)-Wide AND-OR | Programmable | 8 Bidirectional 2 Dedicated | 24 |

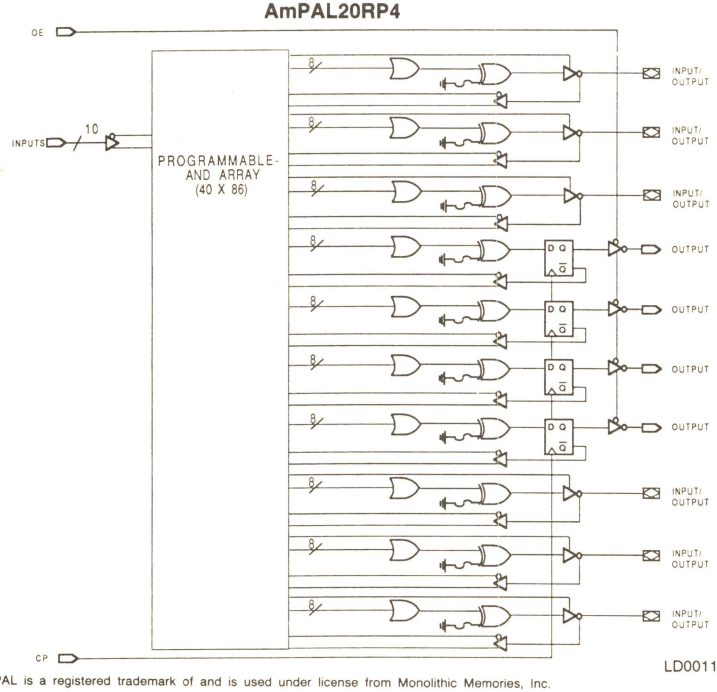
## Block Diagrams

### AmPAL22P10




LD001100

### AmPAL20RP4

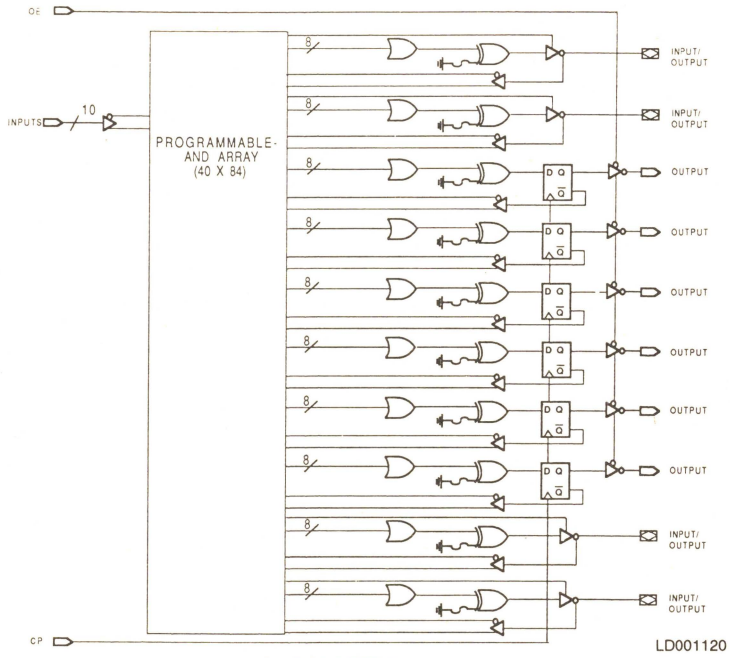



LD001110

*PAL is a registered trademark of and is used under license from Monolithic Memories, Inc.

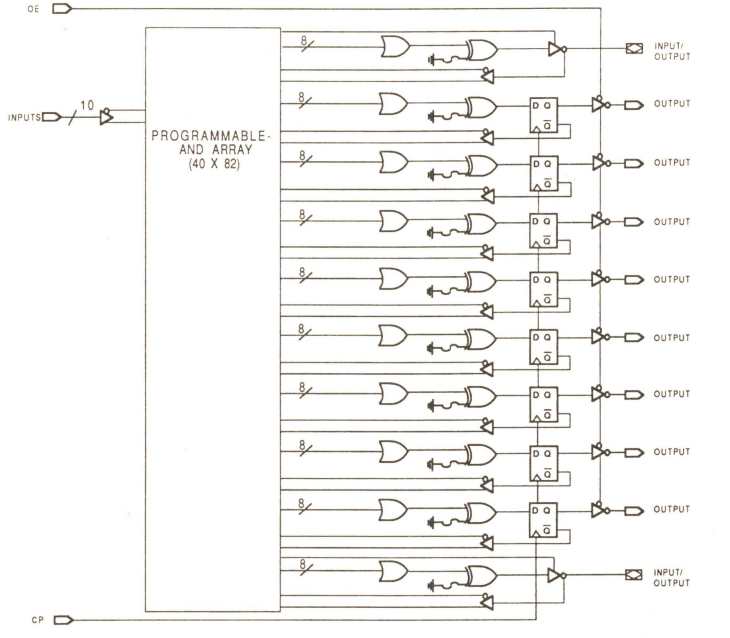
## Block Diagrams (Cont'd.)

### AmPAL20RP6




LD001120

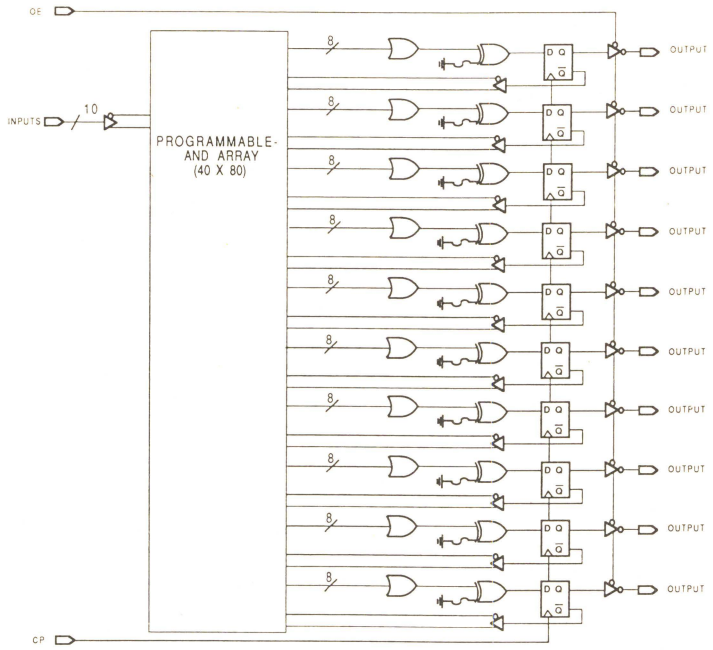
### AmPAL20RP8



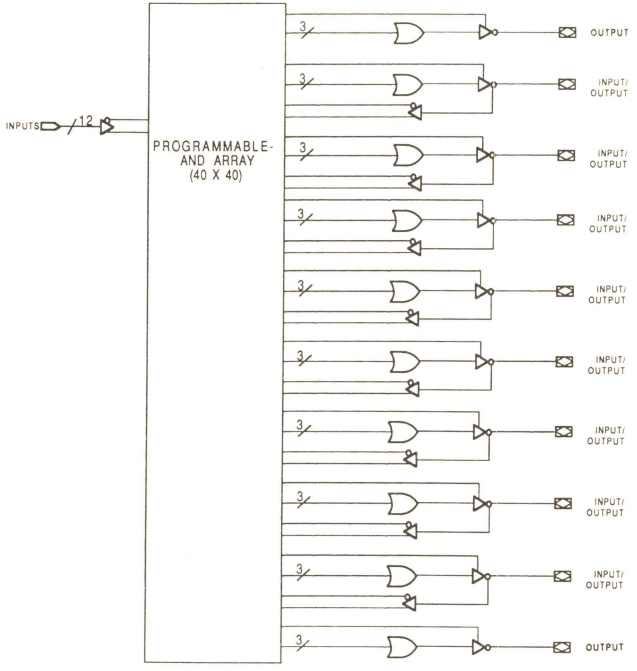

LD001130

## Block Diagrams (Cont'd.)

### AmPAL20RP10




### AmPAL20L10

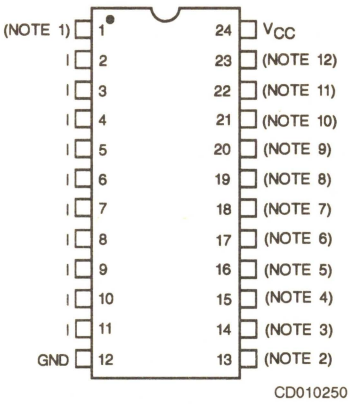



LD001150

## Connection Diagrams

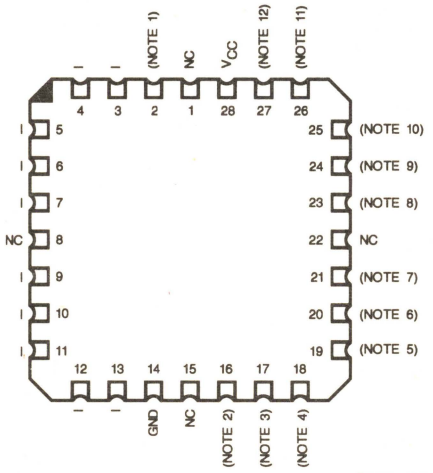
Top View

DIPs*




CD010250

LCC**




CD010260

Note: Pin 1 is marked for orientation.

| Notes: | | 22P10 | 20RP4 | 20RP6 | 20RP8 | 20RP10 | 20L10 |
|---|---|---|---|---|---|---|---|
| | 1 | I | CLK | CLK | CLK | CLK | I |
| | 2 | I | OE | OE | OE | OE | I |
| | 3 | I/O | I/O | I/O | I/O | O | O |
| | 4 | I/O | I/O | I/O | O | O | I/O |
| | 5 | I/O | I/O | O | O | O | I/O |
| | 6 | I/O | O | O | O | O | I/O |
| | 7 | I/O | O | O | O | O | I/O |
| | 8 | I/O | O | O | O | O | I/O |
| | 9 | I/O | O | O | O | O | I/O |
| | 10 | I/O | I/O | O | O | O | I/O |
| | 11 | I/O | I/O | I/O | O | O | I/O |
| | 12 | I/O | I/O | I/O | I/O | O | O |

*Also available in 24-Pin Ceramic Flatpack. Pinouts identical to DIPs.
**Also available in 28-Pin Plastic Leaded Chip Carrier. Pinouts identical to LCC.

## Pin Designations

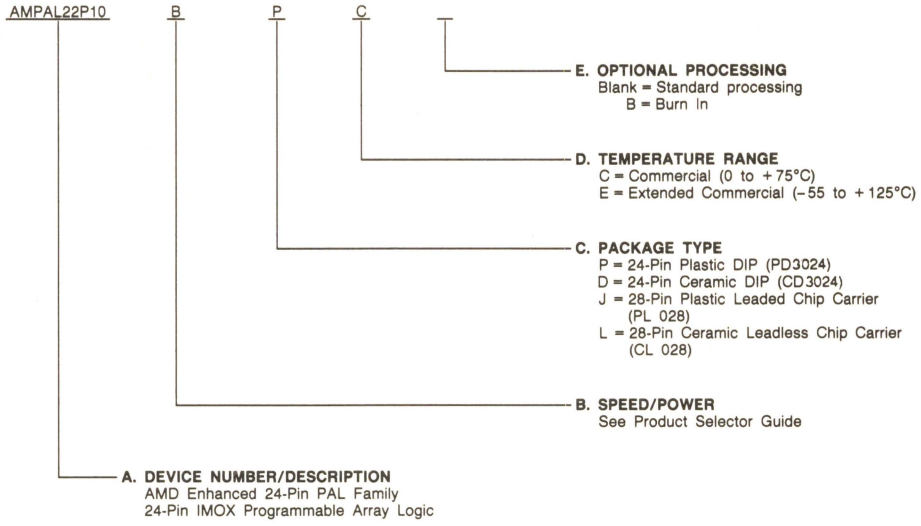
I = Input
I/O = Input/Output
O = Output
$V_{CC}$ = Supply Voltage
GND = Ground
CLK = Clock
OE = Output Enable
NC = No Connect

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AMPAL22P10B/A/AL | PC, DC, DCB, DE, JC, LC, LE |
| AMPAL20RP4B/A/AL | |
| AMPAL20RP6B/A/AL | |
| AMPAL20RP8B/A/AL | |
| AMPAL20RP10B/A/AL | |
| AMPAL20L10B/-20/AL | |

**Valid Combinations**

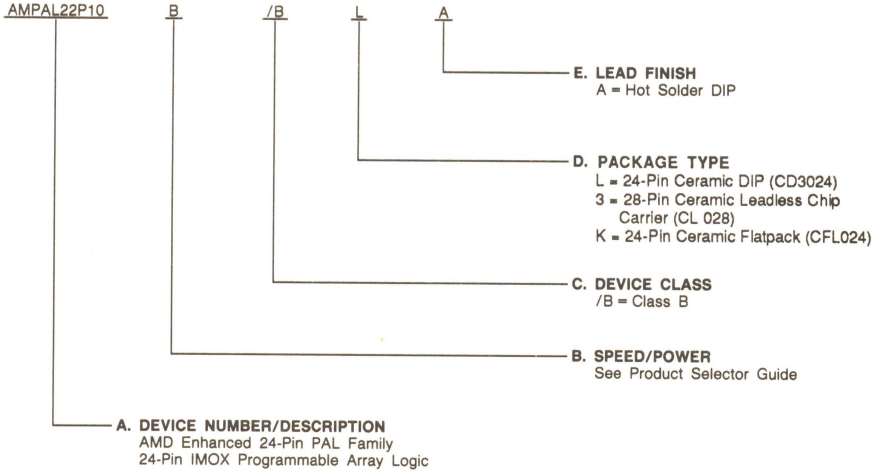
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

## Ordering Information (Cont'd.)

### APL Products

AMD products for Aereospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

**A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AMPAL22P10B/A/AL | /BLA, /B3A, /BKA |
| AMPAL20RP4B/A/AL | |
| AMPAL20RP6B/A/AL | |
| AMPAL20RP8B/A/AL | |
| AMPAL20RP10B/A/AL | |
| AMPAL20L10B/-25/AL | |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

**Group A Tests**

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, & 11

5

## Functional Description

### AMD Enhanced 24-Pin PAL Family Characteristics

All members of the AMD Enhanced 24-Pin PAL Family have common electrical characteristics and programming procedures. All parts are produced with a fusible link at each input to the AND gate array, and connections may be selectively removed by applying appropriate voltages to the circuit.

Initially the AND gates are connected, via fuses, to both the true and complement of each input. By selective programming of fuses the AND gates may be "connected" to only the true input (by blowing the complement fuse), to only the complement input (by blowing the true fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the true and complement fuses are left intact a logical false results on the output of the AND gate, while all fuses blown results in a logical true state. For combinatorial outputs, the AND gates are connected to fixed-OR gates whose outputs become device outputs. For registered outputs, the AND gates are connected to fixed-OR gates whose outputs become output register inputs.

All parts are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are pre-programmed during manufacturing to insure extremely high field programming yields ( > 98%), and provide extra test paths to achieve excellent parametric correlation.

### Power-Up Reset

The registered devices in the AMD PAL family have been designed to reset during system power-up. Following power-up, all registers will be initialized to zero, setting all the outputs to a logic 1. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

### PRELOAD

AMD PAL devices are designed with unique PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the registered output of an AMD PAL device.

A typical functional test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

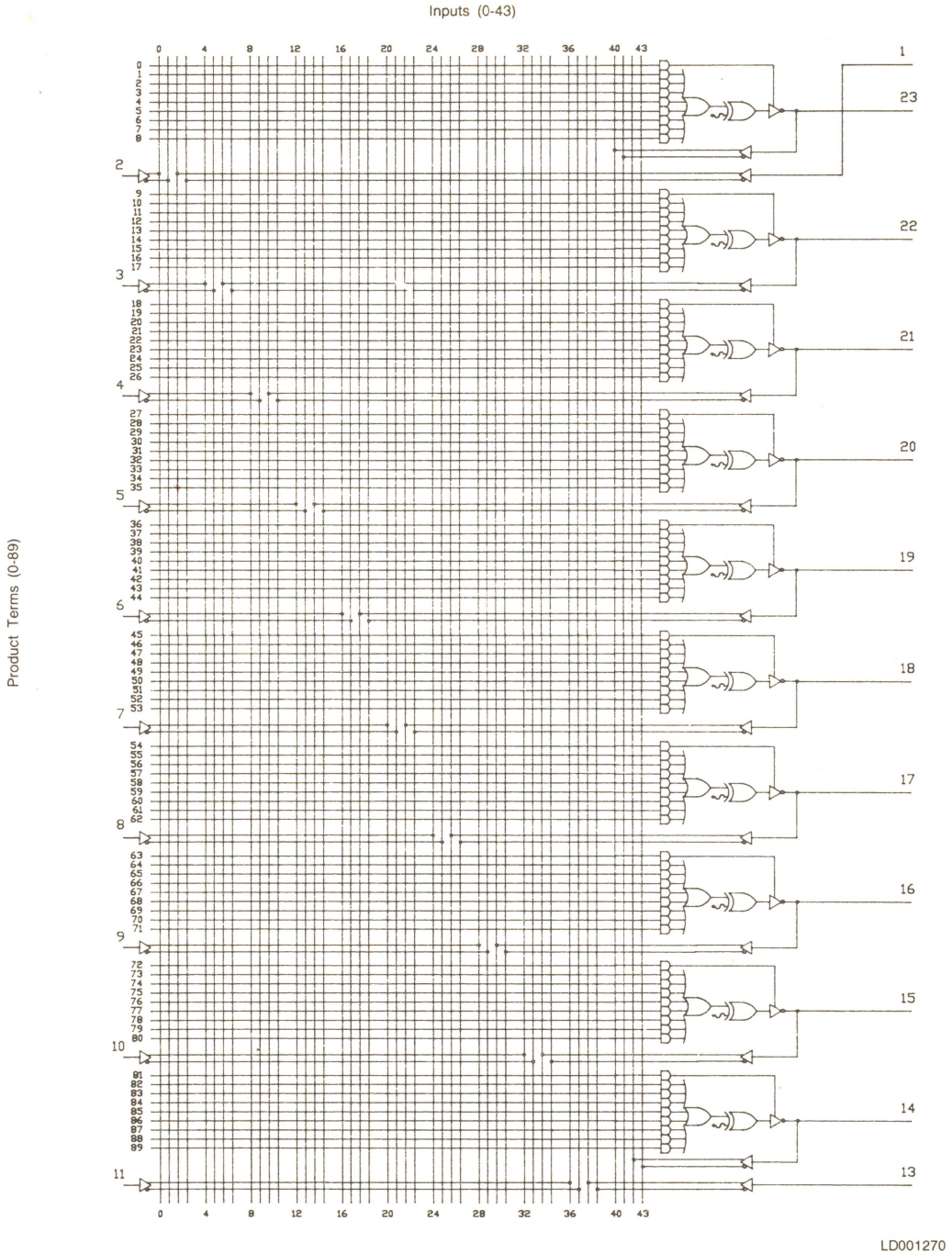



Figure 1. AmPAL22P10 Logic Diagram

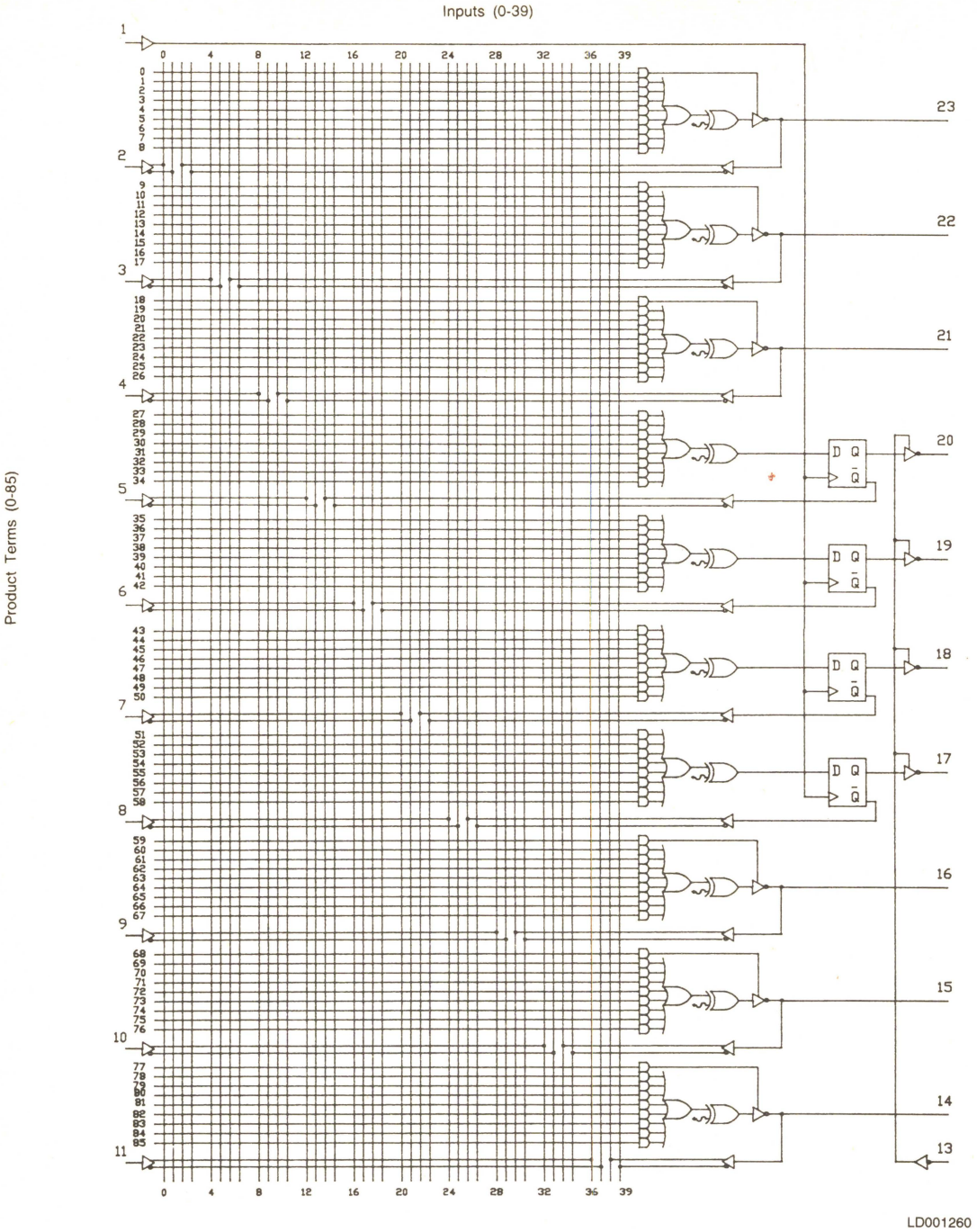



Figure 2. AmPAL20RP4 Logic Diagram

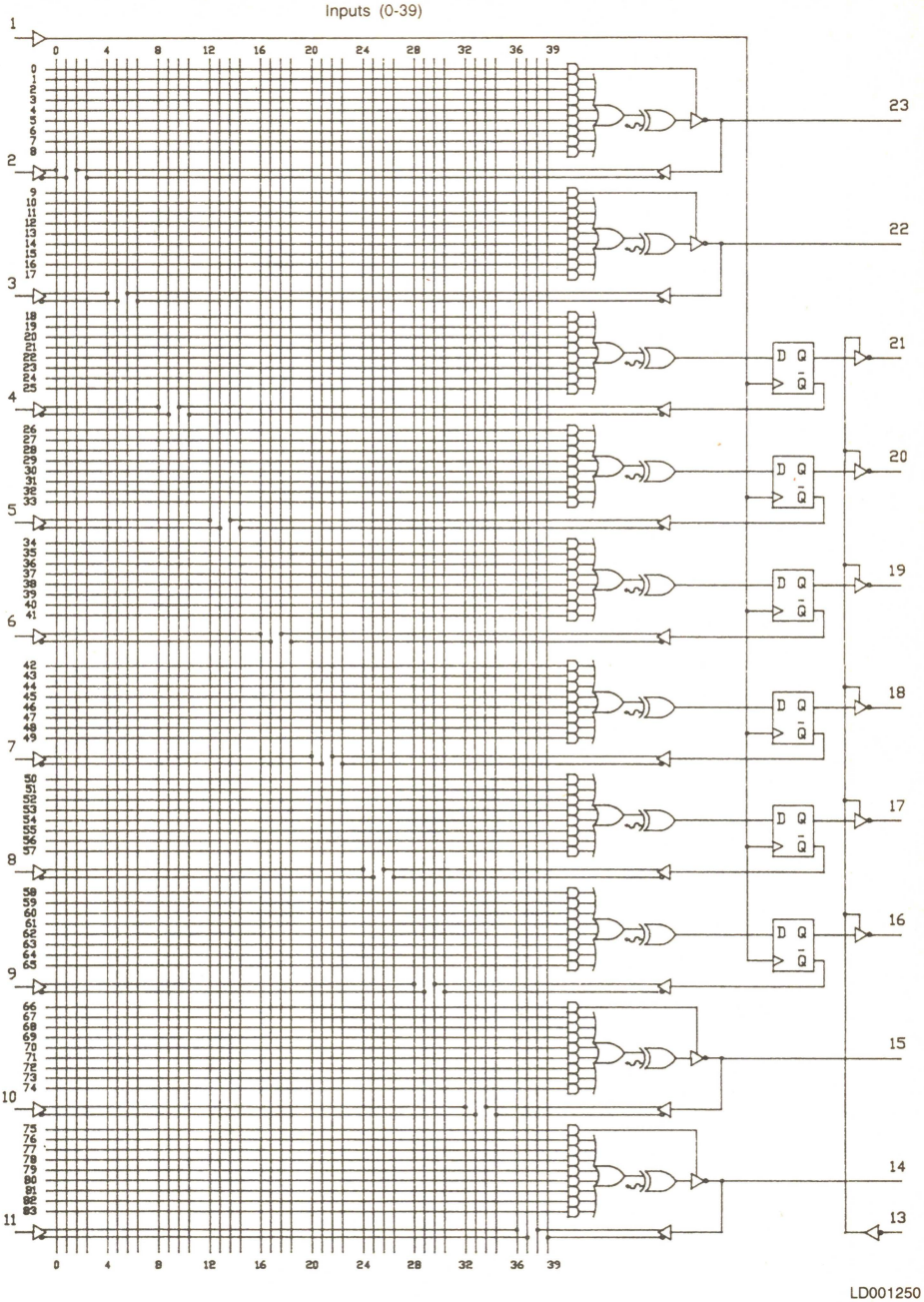



Figure 3. AmPAL20RP6 Logic Diagram

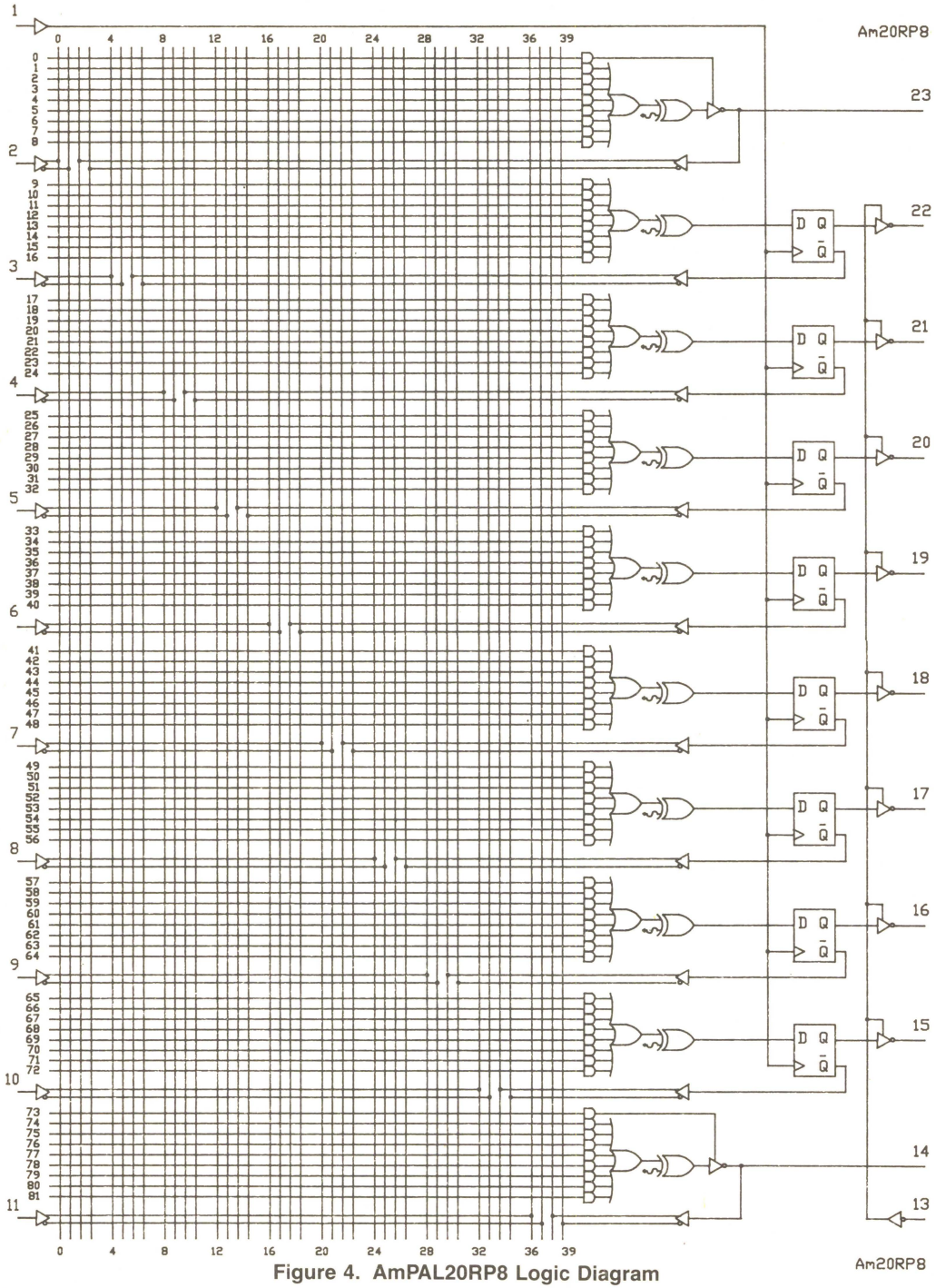



Figure 4. AmPAL20RP8 Logic Diagram

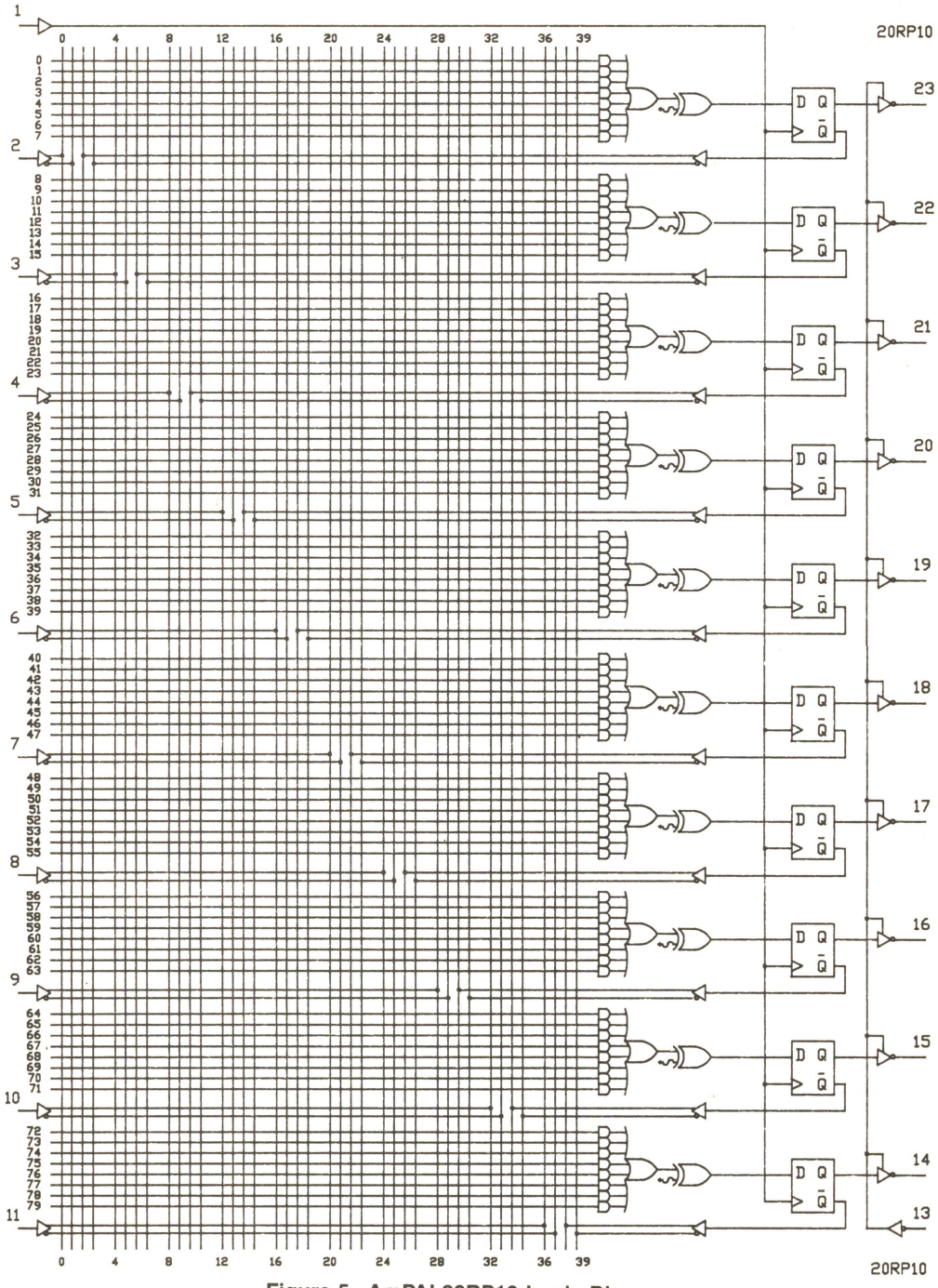
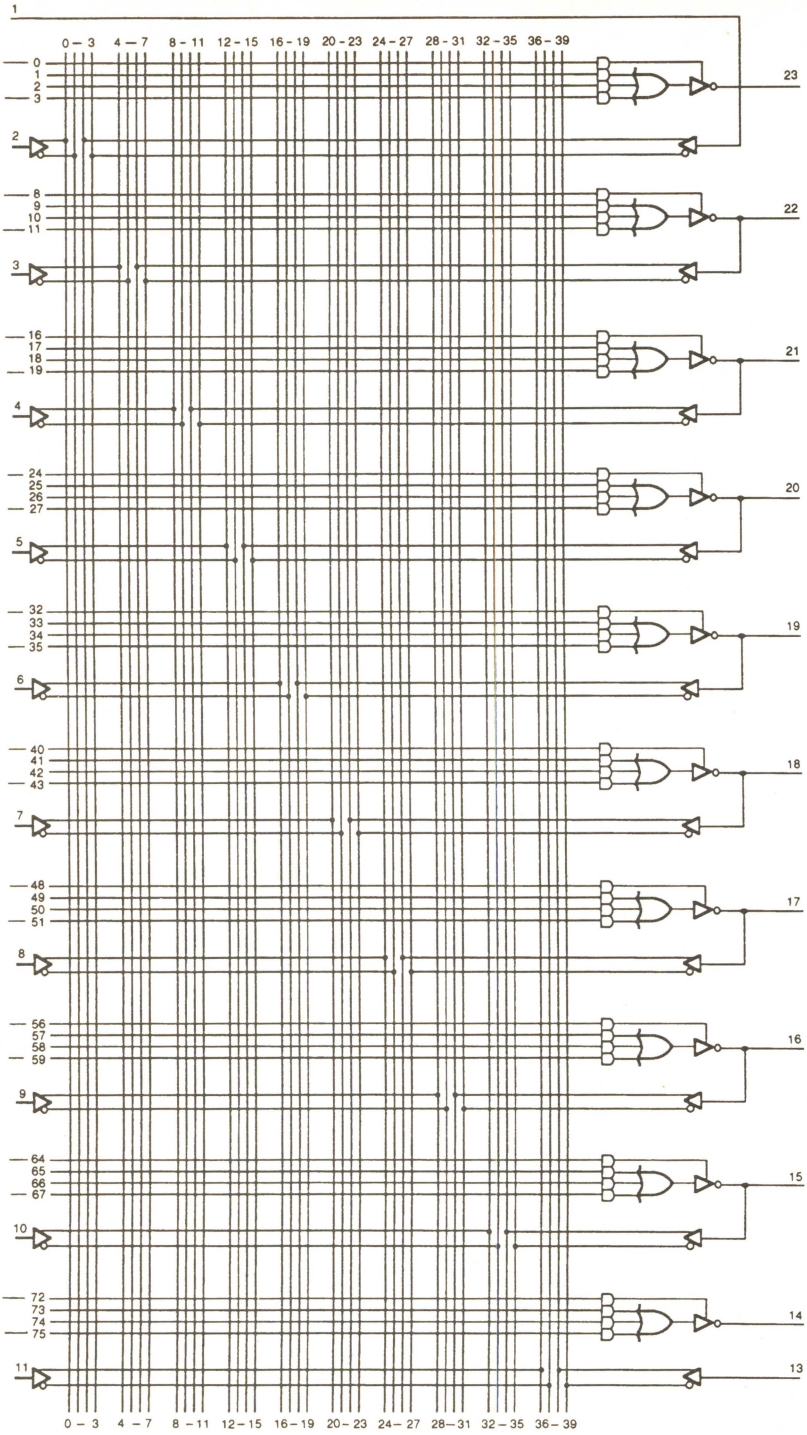



Figure 5. AmPAL20RP10 Logic Diagram



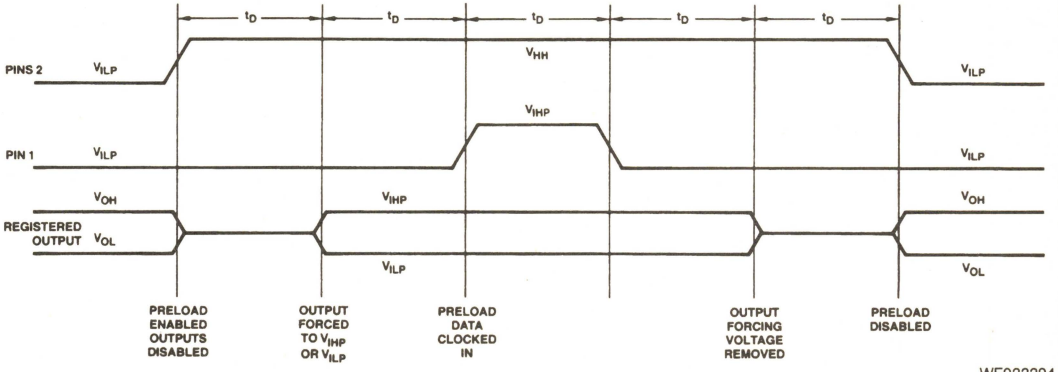

LD001210

Figure 6. AmPAL20L10 Logic Diagram

## PRELOAD of Registered Outputs

The AMD Enhanced 24-pin PAL devices incorporate circuitry to allow loading each register synchronously to either a HIGH or LOW state. This feature simplifies testing since any initial state for the registers can be set to optimize test sequencing.

The pin levels and timing necessary to perform the PRELOAD function are detailed below:




WF022294

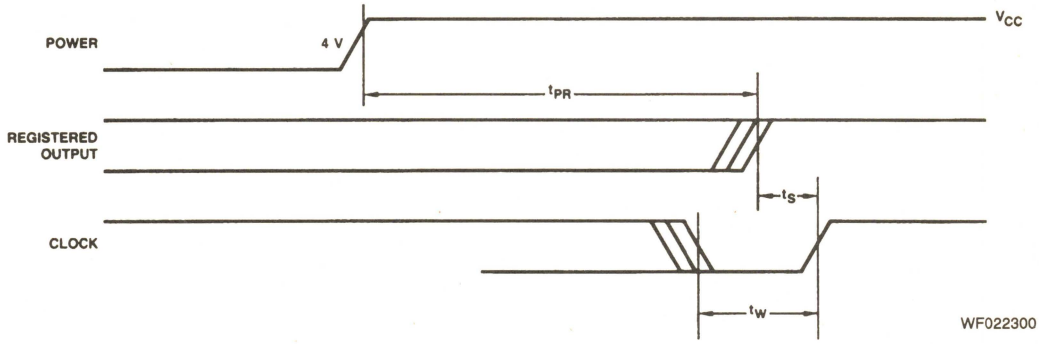
| Par. | Min. | Max. |
|---|---|---|
| $V_{HH}$ | 10 | 12 |
| $V_{ILP}$ | 0 | 0.5 |
| $V_{IHP}$ | 2.4 | 5.5 |

| Level forced on registered output pin during PRELOAD cycle | Register Q output state after cycle |
|---|---|
| $V_{IHP}$ | HIGH |
| $V_{ILP}$ | LOW |

## Power-Up Reset

The registered devices in the AMD Enhanced 24-Pin PAL Family have been designed with the capability to reset during system power-up. Following power-up, all registers will be reset to LOW. The output state will be HIGH. This feature provides flexibility to the designer and is especially valuable in simplifying state-machine initialization. A timing diagram and parameter table are shown below. Due to the asynchronous operation of the power-up RESET and the wide range of ways $V_{CC}$ can rise to its steady state, two conditions are required to ensure a valid power-up RESET. These conditions are:

1. The $V_{CC}$ rise must be monotonic.
2. Following reset, the clock input must not be driven from LOW to HIGH until all applicable input and feedback setup times are met.




WF022300

| Parameters | Description | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| $t_{PR}$ | Power-Up Reset Time | | 600 | 1000 | ns |
| $t_S$ | Input or Feedback Setup Time | See Switching Characteristics | | | |
| $t_W$ | Clock Width | | | | |

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
(Pin 24 to Pin 12) Continuous ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs
(Except During Programming).....−0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 16 V
Output Current Into Outputs During
Programming (Max Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 to +5 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended pericds may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) ..... 0 to +75°C
Supply Voltage ($V_{CC}$) ..... +4.75 to +5.25 V

Extended Commercial (E) Devices
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

Military (M) Devices*
Temperature ($T_A$) ..... −55°C Min.
Temperature ($T_C$) ..... +125°C Max.
Supply Voltage ($V_{CC}$) ..... +4.50 to +5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameter Symbol | Parameter Description | Test Conditions | | | | Min. | Typ. (Note 1) | Max. | Units |
|---|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.2 mA | COM'L | | 2.4 | 3.5 | | V |
| | | | $I_{OH}$ = −2 mA | MIL | | | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 24 mA | COM'L | | | | 0.5 | V |
| | | | $I_{OL}$ = 12 mA | MIL | | | | | |
| $V_{IH}$ (Note 2) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | | 2.0 | | 5.5 | V |
| $V_{IL}$ (Note 2) | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs | | | | | | 0.8 | V |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.40 V | | | | | −20 | −100 | μA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | | | | | | 25 | μA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | | | 1.0 | mA |
| $I_{SC}$ | Output Short-Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 3) | | | | −30 | −60 | −90 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | 20L10 | COM'L | MIL | | | | mA |
| | | | | B | B | | | 210 | |
| | | | | −20 | −25 | | | 165 | |
| | | | | AL | AL | | | 105 | |
| | | | 22P10, 20RP8, 20RP4, 20RP10 20RP6, | B | B | | | 210 | |
| | | | | A | A | | | 210 | |
| | | | | AL | AL | | | 105 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min., $I_{IN}$ = −18 mA | | | | | −0.9 | −1.2 | V |
| $I_{OZH}$ | Output Leakage Current (Note 4) | $V_{CC}$ = Max., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $V_O$ = 2.7 V | | | | | 100 | μA |
| $I_{OZL}$ | | | $V_O$ = 0.4 V | | | | | −100 | |

Notes: 1. Typical limits are at $V_{CC}$ = 5.0 V and $T_A$ = 25°C.
2. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
3. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5V has been chosen to avoid test problems caused by tester ground degradation.
4. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).

### Capacitance*

| Parameter Symbol | Parameter Description | Test Conditions | | Typ. | Units |
|---|---|---|---|---|---|
| $C_{IN}$ | Input Capacitance | $V_{IN}$ = 2.0 V @ f = 1 MHz | Pins 1, 13 | 11 | pF |
| | | | Others | 6 | |
| $C_{OUT}$ | Output Capacitance | $V_{OUT}$ = 2.0 V @ f = 1 MHz | | 9 | |

*These parameters are not 100% tested, but are evaluated at initial characterization and at any time the design is modified where capacitance may be affected.

## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

### Commercial Range

| No. | Parameter Symbol | Parameter Description | "B" Version | | | "A" & "AL" Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 22P10, 20RP4, 20RP6, 20RP8 | | | 15 | | | 25 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 22P10, 20RP4, 20RP6, 20RP8 | | | 18 | | | 25 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 22P10, 20RP4, 20RP6, 20RP8 | | | 15 | | | 25 | ns |
| 4 | $t_{PZX}$ | Pin 13 to Output Enable 20RP4, 20RP6, 20RP8, 20RP10 | | | 15 | | | 20 | ns |
| 5 | $t_{PXZ}$ | Pin 13 to Output Disable 20RP4, 20RP6, 20RP8, 20RP10 | | | 12 | | | 20 | ns |
| 6 | $t_{CO}$ | Clock to Output 20RP4, 20RP6, 20RP8, 20RP10 | | | 12 | | | 15 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 20RP4, 20RP6, 20RP8, 20RP10 | 15 | | | 25 | | | ns |
| 8 | $t_H$ | Hold Time 20RP4, 20RP6, 20RP8, 20RP10 | 0 | | | 0 | | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 27 | | | 40 | | | ns |
| 10 | $t_{WL}/t_{WH}$ | Clock Width | 10/12 | | | 15/15 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 37.0 | | | 25.0 | MHz |

Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

### Military Range

| No. | Parameter Symbol | Parameter Description | "B" Version | | | "A" & "AL" Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 22P10, 20RP4, 20RP6, 20RP8 | | | 20 | | | 30 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 22P10, 20RP4, 20RP6, 20RP8 | | | 25 | | | 30 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 22P10, 20RP4, 20RP6, 20RP8 | | | 20 | | | 30 | ns |
| 4 | $t_{PZX}$ | Pin 13 to Output Enable 20RP4, 20RP6, 20RP8, 20RP10 | | | 20 | | | 25 | ns |
| 5 | $t_{PXZ}$ | Pin 13 to Output Disable 20RP4, 20RP6, 20RP8, 20RP10 | | | 20 | | | 25 | ns |
| 6 | $t_{CO}$ | Clock to Output 20RP4, 20RP6, 20RP8, 20RP10 | | | 15 | | | 20 | ns |
| 7 | $t_S$ | Input or Feedback Setup Time 20RP4, 20RP6, 20RP8, 20RP10 | 20 | | | 30 | | | ns |
| 8 | $t_H$ | Hold Time 20RP4, 20RP6, 20RP8, 20RP10 | 0 | | | 0 | | | ns |
| 9 | $t_P$ | Clock Period ($t_S + t_{CO}$) | 35 | | | 50 | | | ns |
| 10 | $t_{WL}/t_{WH}$ | Clock Width | 12/12 | | | 20/20 | | | ns |
| 11 | $f_{MAX.}$ | Maximum Frequency | | | 28.6 | | | 20.0 | MHz |

Notes: 1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.

## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

### Commercial Range

| No. | Parameter Symbol | Parameter Description | B Versions | | | –25 Versions (Note 4) | | | AL Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 20L10 | | | 15 | | | 20 | | | 25 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 20L10 | | | 18 | | | 20 | | | 25 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 20L10 | | | 15 | | | 20 | | | 25 | ns |

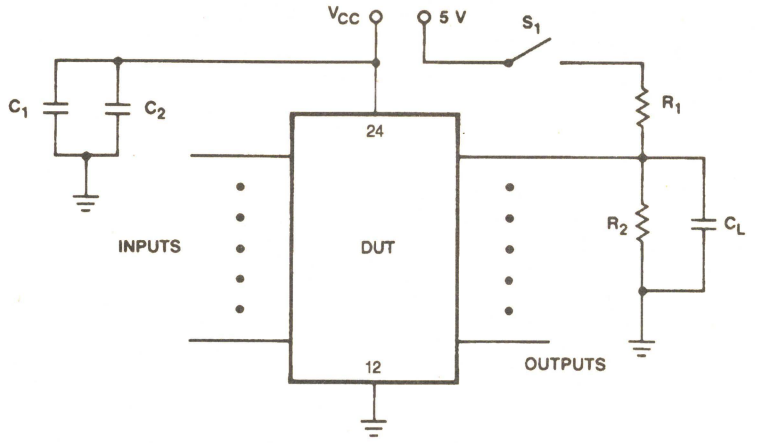
Notes:
1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.
4. AmPAL20L10 only.

### Military Range

| No. | Parameter Symbol | Parameter Description | B Versions | | | –25 Versions (Note 4) | | | AL Versions | | | Units |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | Min. | Typ. (Note 1) | Max. | |
| 1 | $t_{PD}$ | Input or Feedback to Non-Registered Output 20L10 | | | 20 | | | 25 | | | 30 | ns |
| 2 | $t_{EA}$ | Input to Output Enable 20L10 | | | 25 | | | 25 | | | 30 | ns |
| 3 | $t_{ER}$ | Input to Output Disable 20L10 | | | 20 | | | 25 | | | 30 | ns |

Notes:
1. Typical limits are at $V_{CC} = 5.0$ V and $T_A = 25°C$.
2. $t_{PD}$ is tested with switch $S_1$ closed and $C_L = 50$ pF.
3. For three-state outputs, output enable times are tested with $C_L = 50$ pF to the 1.5 V level; $S_1$ is open for high impedance to HIGH tests and closed for high impedance to LOW tests. Output disable times are tested with $C_L = 5$ pF. HIGH to high impedance tests are made to an output voltage of $V_{OH} - 0.5$ V with $S_1$ open; LOW to high impedance tests are made to the $V_{OL} + 0.5$ V level with $S_1$ closed.
4. AmPAL20L10 only.

## Switching Test Circuit



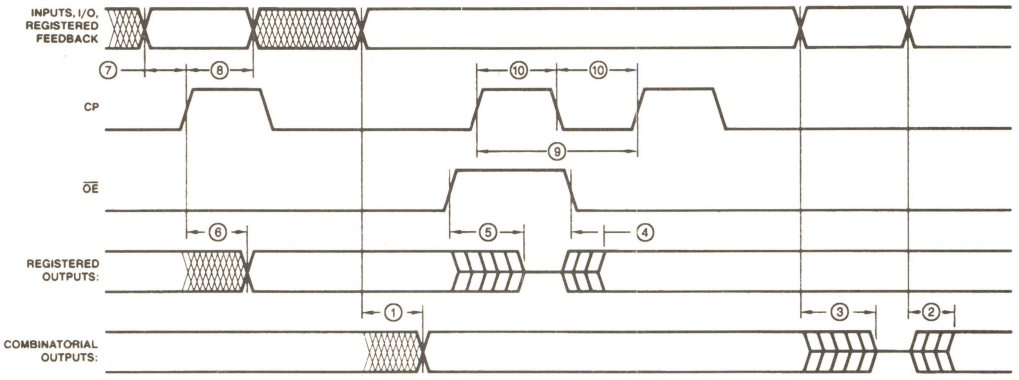

TC003051

Note: $C_1$ and $C_2$ are to bypass $V_{CC}$ to ground.

| TEST OUTPUT LOADS | | |
|---|---|---|
| **Name** | **Commercial** | **Military** |
| $R_1$ | 200 Ω | 390 Ω |
| $R_2$ | 390 Ω | 750 Ω |
| $C_1$ | 1 μF | 1 μF |
| $C_2$ | 0.1 μF | 0.1 μF |
| $C_L$ | 50 pF | 50 pF |

5

## Switching Waveforms



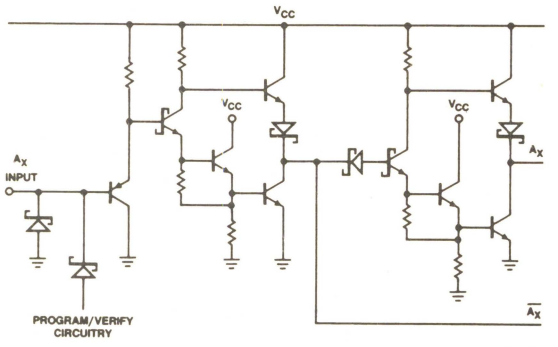

WF002571

## Key to Timing Diagram

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| | MUST BE STEADY | WILL BE STEADY |
| | MAY CHANGE FROM H TO L | WILL BE CHANGING FROM H TO L |
| | MAY CHANGE FROM L TO H | WILL BE CHANGING FROM L TO H |
| | DON'T CARE; ANY CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| | DOES NOT APPLY | CENTER LINE IS HIGH IMPEDANCE "OFF" STATE |

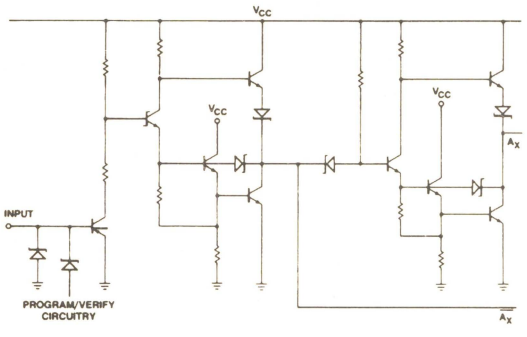
KS000010

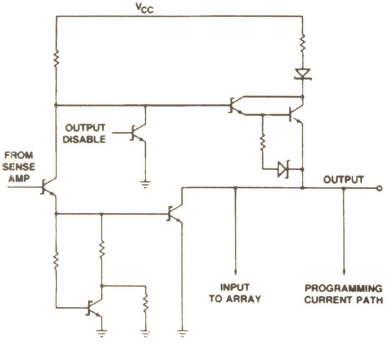
## Input Circuitry




IC000720

## Output Circuitry




IC000801

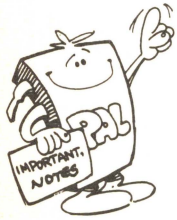
### Security Fuse Programming

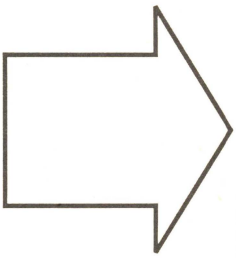
A single fuse is provided on each device to prevent unauthorized copying of PAL device fuse patterns. Once blown, the circuitry enabling fuse verification and registered output PRELOAD is premanently disabled.

Programming of the security fuse is the same as an array fuse. Verification of a blown security fuse is accomplished by verifying the whole fuse array as if every fuse is blown.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

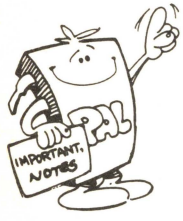

IMPORTANT
NOTES



## Data Sheets

- TTL/CMOS PAL Devices
- TTL/CMOS AmPAL Devices
- PROSE/PLS Sequencers
- FPC/PEG Sequencers
- ECL PAL Devices
- HAL/ZHAL Devices
- Military PAL Devices
- Logic Cell Array
- Electrical Definitions

5


IMPORTANT
NOTES

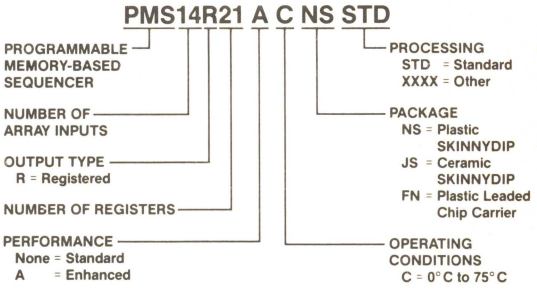
# PROSE™
# Programmable Sequencer

# PMS14R21/A

## Features/Benefits

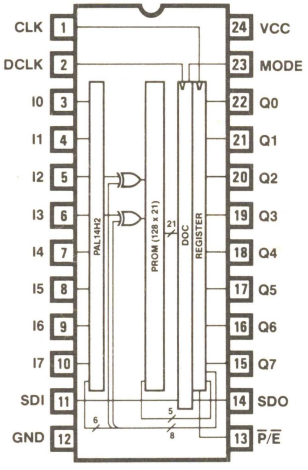
- User-programmable synchronous state machine
- 30 MHz maximum internal frequency, 25 MHz external
- Allows up to 128 states for complex designs
- 8 inputs and 8 outputs with 13 buried feedback signals
- PAL® array optimizes input decoding
- Four-way branching in one cycle
- User-selectable asynchronous preset or enable saves pins
- Power-up preset for start-up in known state
- Diagnostics-On-Chip™ shadow register eases chip and board testing by allowing preload and observation of output register
- Supported by PALASM®2 software and other development tools
- Programmable on standard logic programmers
- Security fuse prevents pattern duplication
- Space-saving 24-pin SKINNYDIP® and 28-pin PLCC packages

## Ordering Information



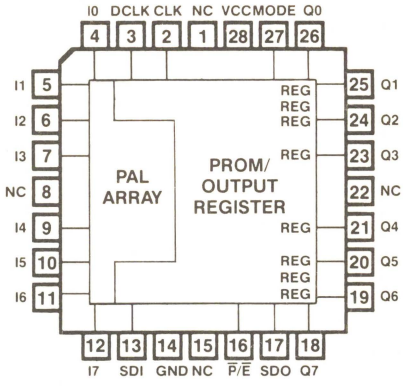

## Pin Configurations

PMS14R21/A




SKINNYDIP Package

PMS14R21/A




Plastic Leaded Chip Carrier

## Package Drawings

(refer to PAL Device Package Outlines)

## Description

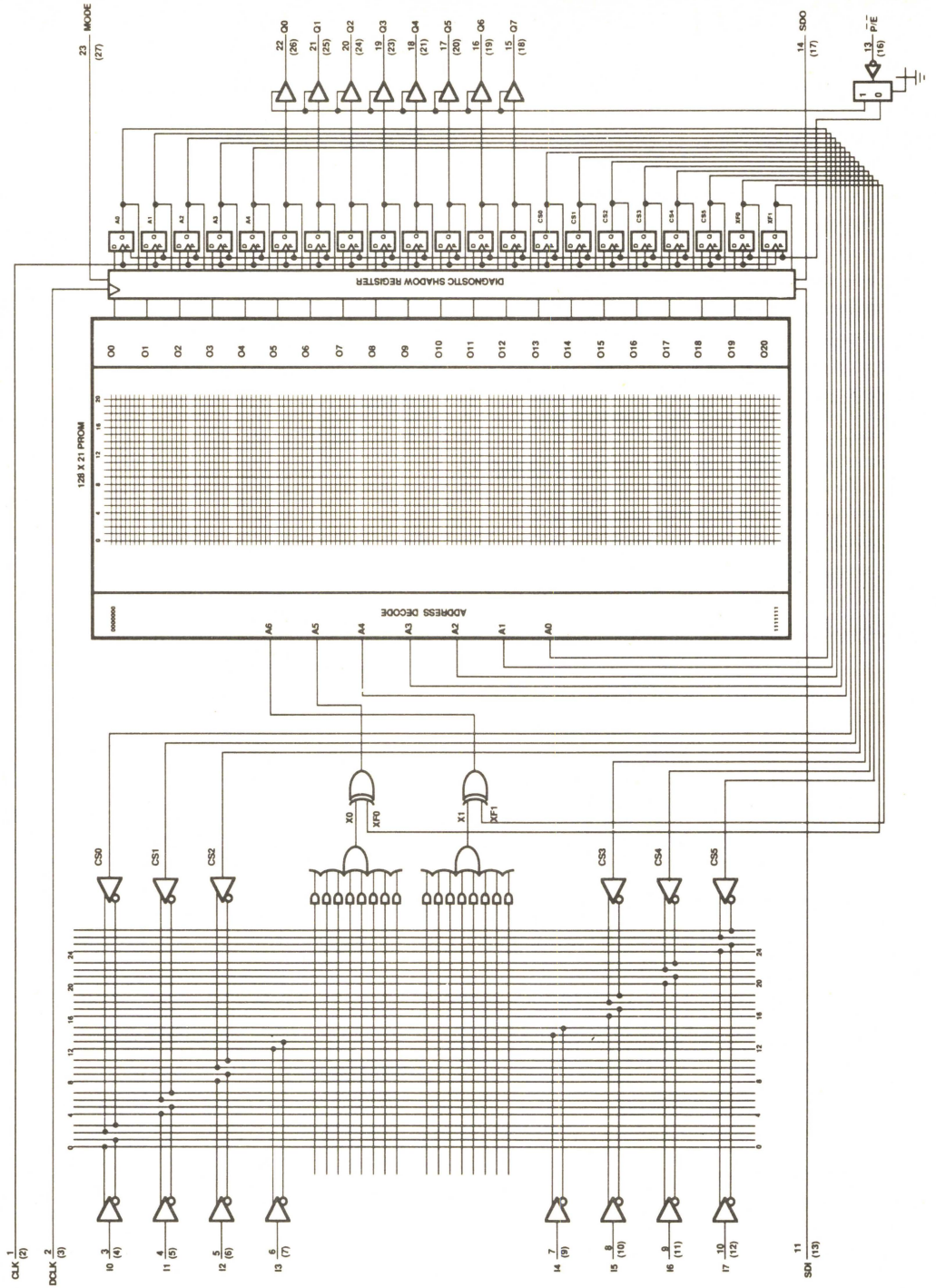
The PMS14R21 combines the basic elements of a sequencer — the conditional input logic and state memory — into one programmable device. A 14H2 PAL array provides programmable input condition decoding while a 128-word by 21-bit PROM array stores the state information and outputs. The combination of both PAL and PROM arrays on a single chip provides very high performance.



10334A
JANUARY 1988

## Logic Diagram

DIP (PLCC) Pinouts



The device has a balanced architecture, with eight inputs for sequencer control and eight outputs for system control. Thirteen internal signals feed back from the state registers to control next state selection. The next state is selected by the five primary address signals and the two branch address signals, allowing up to four-way branching in one cycle.

Two branch control signals are generated by the PAL array as a function of the eight device inputs and six feedback signals. The feedback signals allow programmable condition select decoding on the inputs. The programmable decoding provides efficiency and speed even in complex designs. Exclusive-OR (XOR) gates on the PAL outputs, controlled by two additional feedback lines, help save device resources by allowing merging of states.

A 21-bit output register is paralleled by a 21-bit shadow register. The shadow register is a parallel/serial register that implements Diagnostics-On-Chip. In normal mode, the shadow register does not affect the operation of the state machine. In diagnostic mode, the shadow register allows control and observation of the output register.

The user can select either asynchronous preset or asynchronous output enable, whichever is required for the application. Preset is automatically performed on power up. A security fuse, once programmed, prevents a competitor from reading the PAL array pattern.

## Definition of signals

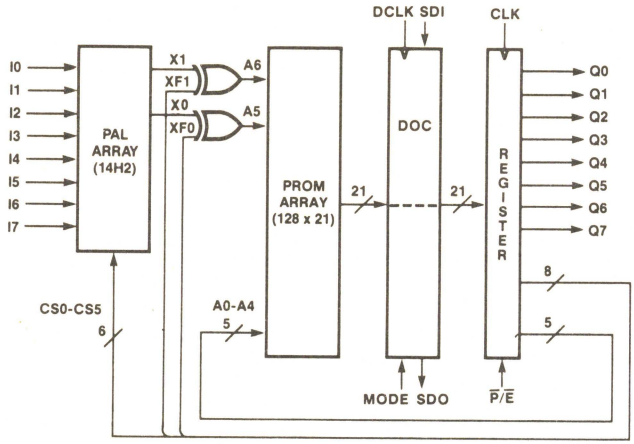
**Pin Signals:**

| | |
|---|---|
| I0-I7 | Inputs to PAL array |
| Q0-Q7 | Registered outputs from PROM |
| $\overline{P}/\overline{E}$ | Programmable asynchronous function pin: default is active low Preset (output register goes high), programmed is active low output enable |
| CLK | Output register clock |
| DCLK | Diagnostic register clock |
| MODE | Diagnostic mode selection |
| SDI | Serial Data Input to diagnostic register |
| SDO | Serial Data Output from diagnostic register |

**Internal Signals:**

| | |
|---|---|
| CS0-CS5 | Condition Select feedback from output register to PAL array |
| X0-X1 | PAL array outputs; inputs to XOR gates |
| XF0-XF1 | XOR gate control feedback from output register |
| A5-A6 | XOR gate outputs providing two highest order address bits to PROM array |
| A0-A4 | Feedback from output register providing five lowest order address bits to PROM array |
| O0-O20 | Output register contents |
| S0-S20 | Shadow diagnostic register contents |

## Block Diagram




5

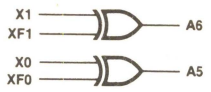
## Architecture Description

Detailed knowledge of the PMS14R21 architecture is not necessary for use of the device; a design is entered into a software development system which automatically optimizes the design and creates a programming file. The user only needs to create a design description for the software to use. The following architecture description is provided for better knowledge of the device, but the software automatically handles the features described.

The following sections of the PROSE architecture are described in detail:

- PAL array
- Exclusive-OR (XOR) gates
- PROM array
- Output register
- Feedback to PAL array (CS0-CS5)
- Feedback to XOR gates (XF0-XF1)
- Feedback to PROM array (A0-A4)
- Programmable asynchronous Preset or Enable ($\overline{P}/\overline{E}$)
- Power-up preset
- Security fuse

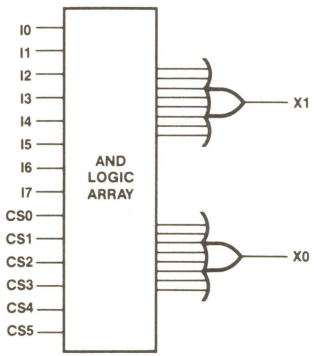
## Exclusive-OR Gates




| X0/1 | XF0/1 | A5/6 |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

The two PAL array outputs (X0-X1) feed two Exclusive-OR (XOR) gates, with the other inputs (XF0-XF1) feeding back from the output register. The XOR gate outputs are the two highest-order address bits to the PROM. The XOR gates allow both the PAL array and the state to control the two address bits to the PROM, which provides four-way branching. This combination role allows the optimum use of device resources.

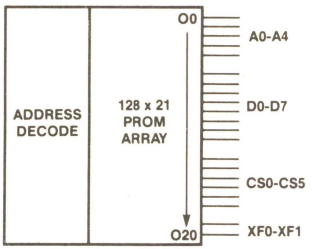
## PAL Array




The PAL array implements the equivalent of a 14H2 circuit with fourteen inputs, two active high outputs, and eight product terms per output. Eight inputs come from pins (I0-I7) while the other six feed back from the output register (CS0-CS5).The outputs (X0-X1) feed two XOR gates.

The PAL array provides conditional input decoding, using the current inputs and state feedback to help control the two highest-order address bits to the PROM. The feedback signals enable the desired function of the inputs to force a branch to a given state. Default branches are handled easily because no product term will be on, providing 00 for X0-X1. Unconditional branches can be handled the same way, with no product terms on.

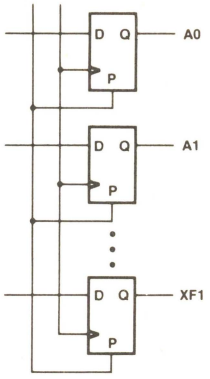
## PROM Array




The PROM array is organized 128x21, or 128 words of 21 bits each, for a total of 2688 bits. The address lines are A0-A6, including A0-A4 from the output register and A5-A6 from the XOR gates. The PROM outputs (O0-O20) feed the 21-bit output register.

The PROM stores the state information just as a discrete PROM would in a discrete sequencer design. The 128 locations provide up to 128 possible states in an optimum design. The PROM outputs include eight fed to output pins for system control. The architecture directly implements a Moore machine, although a Mealy machine can be created by delaying the outputs from the state by one cycle. Thirteen PROM outputs feed back for extensive next-state control.

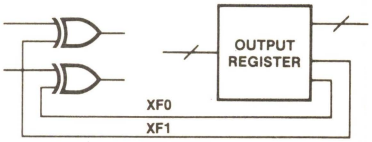
## Output Register




The output register consists of twenty-one D-type flip-flops which are loaded from the PROM on the rising edge of the clock signal, CLK. Eight of the outputs lead to device output pins (Q0-Q7) while the other thirteen are fed back internally (A0-A4, XF0-XF1, CS0-CS5).

The thirteen buried flip-flops help control the sequencer, while the eight dedicated registered outputs provide system control resulting from the sequencer operation.

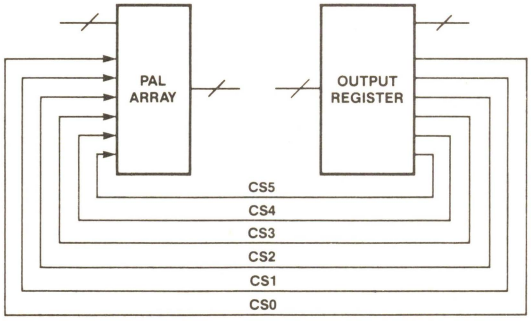
## Feedback to XOR Gates (XF0-XF1)




Two bits of the output register (XF0-XF1) feed back to the XOR gates.

These two feedback signals allow either output of the PAL array to be inverted. This helps software optimization of resources by allowing the same product terms or PAL outputs to be used for different branches.

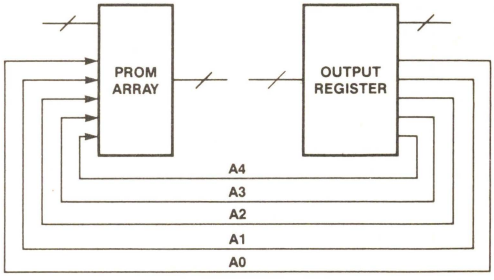
## Feedback to PAL Array (CS0-CS5)




Six bits of the output register feed back to the PAL array. They are used as regular PAL array inputs.

These six Condition Select bits define the function to be performed on the eight external condition inputs. Any Boolean function may be performed, up to the limits of the PAL array architecture. The feedback signals generally select given combinations of inputs (product terms) necessary for branching to next states. For unconditional branches or default branches, they will not select a product term.

## Feedback to PROM Array (A0-A4)

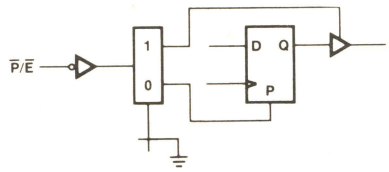



Five bits of the output register feed back to the PROM array. These signals become address lines A0-A4 for the PROM.

The register feedback to the PROM provides primary next state selection based only upon the current state. Address lines A0-A4 define four of the 128 specific locations within the PROM. The actual branch to one of these locations is selected by A5-A6 based on the current inputs.

5

## Programmable Asynchronous Preset or Enable

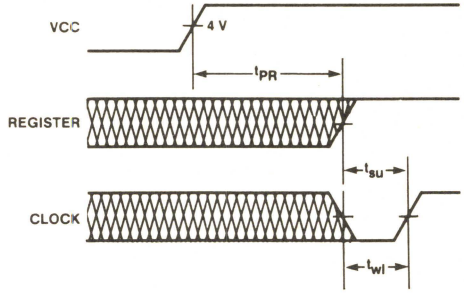



A programmable function pin allows the user to select either asynchronous preset or asynchronous output enable. The choice is programmed into the device when the arrays are programmed. The unprogrammed state is preset.

The programmable function allows the user to choose the function required for the design, saving precious pins. If preset is selected, a low signal on the pin will preset all twenty-one output flip-flops to the high state asynchronously. This is equivalent to the power-up state, and adds the capability of presetting without powering down. The outputs will always be enabled, and act as two-state outputs.

If enable is selected (i.e., the select fuse is programmed), a low signal on the pin will enable the eight output signals (Q0-Q7) from the output register. When the pin is high, the outputs are disabled, and the output pins are in the high-impedance state.

## Power-Up Preset




The twenty-one output flip-flops will power up to the high state. This allows easy initialization, since the starting state is always known. The power-up preset time, $t_{PR}$, is 1 $\mu$s maximum. The required setup time and clock widths are listed in the specifications.

## Security Fuse

After programming and verification, a PMS14R21 design can be secured by programming the security fuse. Once programmed, this fuse defeats readback of the PAL array fuse pattern by a device programmer, making proprietary designs very difficult to copy.

## Absolute Maximum Ratings

| | Operating | Programming |
|---|---|---|
| Supply voltage $V_{CC}$ | -0.5 V to 7.0 V | -0.5 V to 12.0 V |
| Input voltage | -1.5 V to 5.5 V | -1.0 V to 12.0 V |
| Off-state output voltage | 5.5 V | 12.0 V |
| Storage temperature | -65°C to +150°C | |

## Sequencer Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL[1] | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | STD | | | A | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $V_{CC}$ | Supply voltage | | 4.75 | 5 | 5.25 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of CLK | Low | 25 | 15 | | 20 | 15 | | ns |
| | | High | 10 | 5 | | 10 | 5 | | |
| $t_{su}$ | Setup time from I7-I0 to CLK | | 35 | 25 | | 28 | 22 | | ns |
| $t_h$ | Hold time for I7-I0 to CLK | | 0 | -5 | | 0 | -5 | | ns |
| $t_{aw}$ | Asynchronous preset width | | 10 | 5 | | 10 | 5 | | ns |
| $t_{ar}$ | Asynchronous preset recovery | | 20 | 10 | | 20 | 10 | | ns |
| $T_A$ | Operating free-air temperature | | 0 | 25 | 75 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

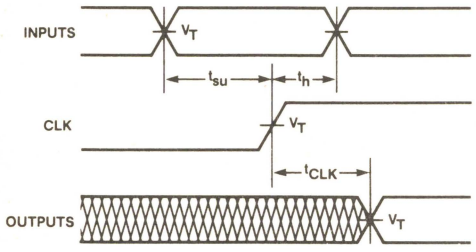
| SYMBOL | PARAMETER | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | 2.0 | | | V |
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = -18 mA | | -0.8 | -1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | -0.02 | -0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | μA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 200 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.3 | 0.45 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -3.2 mA | 2.4 | 3.2 | | V |
| $I_{OZL}$ | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | -100 | μA |
| $I_{OZH}$ | | | $V_O$ = 2.4 V | | | 100 | μA |
| $I_{OS}$[3] | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0 V | -20 | -50 | -90 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 170 | 210 | mA |
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 2.0 V at f = 1 MHz | | | 8 | | pF |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 2.0 V at f = 1 MHz | | | 11 | | pF |
| $C_{CLK}$ | Clock capacitance | $V_{CLK}$ = 2.0 V at f = 1 MHZ | | | 13 | | pF |

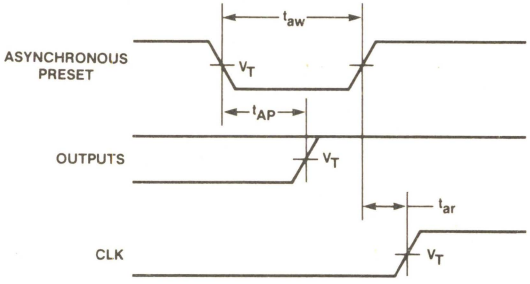
Notes:
1. The PMS14R21/A is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories.
2. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. No more than one output should be shorted at a time and duration of the short-circuit should not exceed one second.

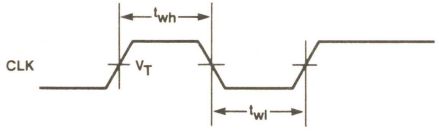
## Sequencer Switching Characteristics Over Operating Conditions

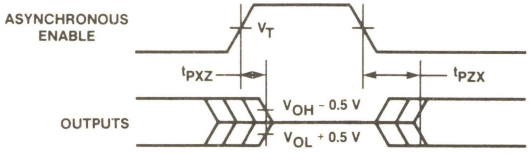
| SYMBOL | PARAMETER | TEST CONDITIONS | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | STD | | | A | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{CLK}$ | CLK to output | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | 10 | 20 | | 9 | 12 | ns |
| $t_{PZX}$ | $\overline{E}$ to output enable | | | 10 | 20 | | 10 | 15 | ns |
| $t_{PXZ}$ | $\overline{E}$ to output disable | | | 10 | 20 | | 10 | 15 | ns |
| $t_{AP}$ | Asynchronous preset to output | | | 14 | 25 | | 14 | 25 | ns |
| $f_{MAX}$ | Maximum frequency, external ($1/(t_{su} + t_{CLK})$) | | 18 | 25 | | 25 | 30 | | MHz |
| | Maximum frequency, internal | | 25 | 30 | | 30 | 35 | | |

## Sequencer Switching Waveforms



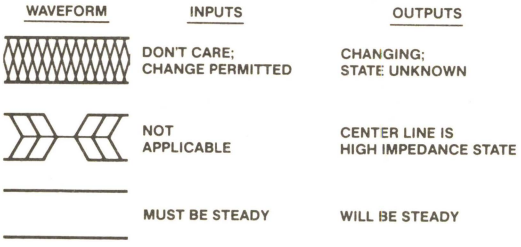



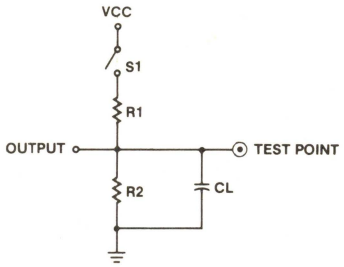




Notes:
1. $V_T$ = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Key to Timing Diagram

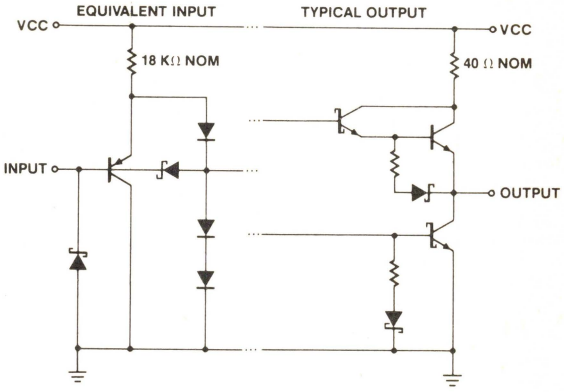


## Switching Test Load




Notes:
1. Propagation delays are tested with switch $S_1$ closed. $C_L$ = 30 pF and measured at 1.5 V output level.
2. $t_{PZX}$ is measured at the 1.5 V output level with $C_L$ = 30 pF. $S_1$ is open for high impedance to "1" test, and closed for high impedance to "0" test.
3. $t_{PXZ}$ is tested with $C_L$ = 5 pF. S1 is open for "1" to high impedance test, measured at $V_{OH}$ -0.5 V output level: $S_1$ is closed for "0" to high impedance test measured at $V_{OL}$ +0.5 V output level.

## Schematic of Inputs and Outputs



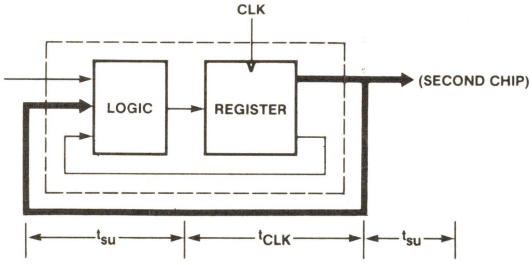

## fMAX Parameters

The parameter fMAX is the maximum clock rate at which the device is guaranteed to operate. Because flexibility inherent in programmable logic devices offers a choice of clocked flip-flop designs, fMAX is specified for two types of synchronous designs.

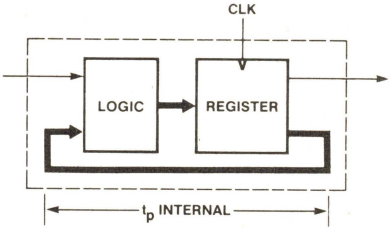
The first type of design is a state machine with feedback signals sent off-chip. This external feedback could go back to the device inputs, or to a second device in a multi-chip state machine. The slowest path defining the period is the sum of the clock-to-output time and the input setup time for the external signals. The reciprocal, fMAX, is the maximum frequency with external feedback or in conjunction with an equivalent speed device. This fMAX is designated "fMAX external."

The second type of design is a single-chip state machine with internal feedback only. In this case, flip-flop inputs are defined by the device inputs and flip-flop outputs. Under these conditions, the period is limited by the internal delay from the flip-flop outputs through the internal feedback and logic to the flip-flop inputs. This fMAX is designated "fMAX internal."

5




$f_{MAX}$ EXTERNAL; $1/(t_{CLK} + t_{su})$




$f_{MAX}$ INTERNAL; $1/(t_p$ INTERNAL)

## Diagnostics Function Table

| INPUTS | | | | OUTPUTS | | | OPERATION |
|---|---|---|---|---|---|---|---|
| MODE | SDI | CLK | DCLK | $Q_{20}$-$Q_0$ | $S_{20}$-$S_0$ | SDO | |
| L | X | ↑ | * | Qn ← PROM | HOLD | $S_{20}$ | Load output register from PROM array |
| L | X | * | ↑ | HOLD | $S_n \leftarrow S_{n-1}$<br>$S_0 \leftarrow$ SDI | $S_{20}$ | Shift shadow register data |
| L | X | ↑ | ↑ | Qn ← PROM | $S_n \leftarrow S_{n-1}$<br>$S_0 \leftarrow$ SDI | $S_{20}$ | Load output register from PROM array while shifting shadow register data |
| H | X | ↑ | * | $Q_n \leftarrow S_n$ | HOLD | SDI | Load output register from shadow register |
| H | L | * | ↑ | HOLD | $S_n \leftarrow Q_n$ | SDI | Load shadow register from output register |
| H | L | ↑ | ↑ | $Q_n \leftarrow S_n$ | $S_n \leftarrow Q_n$ | SDI | Swap output and shadow registers |
| H | H | * | ↑ | HOLD | HOLD | SDI | No operation† |

* Clock must be steady or falling.

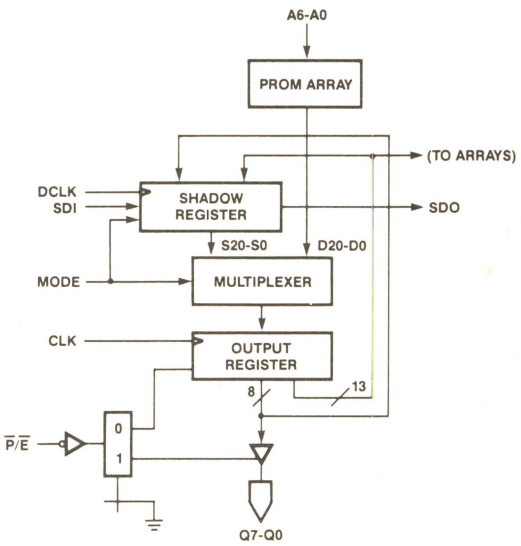
† Reserved operation for 54/74S818 8-Bit Diagnostic Register.

## Diagnostics-On-Chip

Diagnostics-On-Chip, or DOC™, is a test method that allows complete controllability and observability of a device, which results in complete testability. The DOC circuitry includes the serial diagnostic register and parallel connections to and from the output register.

The diagnostic (or shadow) register is a 21-bit serial/parallel register. It has its own clock, called DCLK. It can shift serially, or parallel transfer its contents to or from the output register. The MODE input selects serial or parallel mode (low or high, respectively). The serial shift input is SDI and the output is SDO. SDI doubles as a control input if not in serial mode (MODE is high), and is then transferred directly to SDO. See the function table above for details.

## Diagnostics Block Diagram



### Diagnostic Test Sequence

A typical diagnostic test sequence would proceed as follows. First, a test vector would be serially shifted into the diagnostic register. This is done as shown in the waveforms under Shifting Diagnostic Register, with MODE set low and clocking DCLK. During shifting, the sequencer operates completely independently.

The test vector is then transferred to the output register by switching MODE to high and clocking the output register. This is shown in the waveforms under Loading Output Register from Diagnostic Register. This function makes the output register controllable. MODE goes back low and the system is cycled with the test vector as during normal operation.

Test results in the output register are observed by again setting MODE high and clocking the diagnostic clock. This loads the diagnostic register from the output register. This process is shown in the waveforms under Loading Diagnostic Register from Output Register. The results can then be shifted out by switching MODE low and clocking the diagnostic clock. These functions provide complete observability of the test results in the output register. Note that while the test results are being shifted out, a new test vector may be shifted into the diagnostic register.

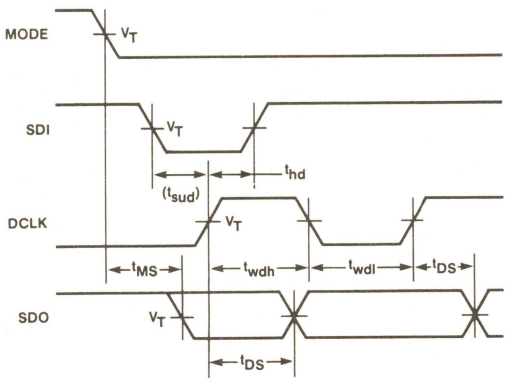
When a test vector is loaded into the output register, the current state of the output register may be required for test analysis or for system restart at the same state. In either case, the output register contents must be transferred to the diagnostic register for saving. Diagnostics-On-Chip allows the registers to swap their contents for this purpose. The CLK and DCLK signals can be separated by no more than $t_{skew}$, as shown in the waveforms under Swap Registers.

A readback function is reserved for the 8-bit Diagnostic Register device. The required signals create a No Operation condition in the PMS14R21 device. This is shown in the waveforms under No Operation.

### Using Diagnostics-On-Chip in a System

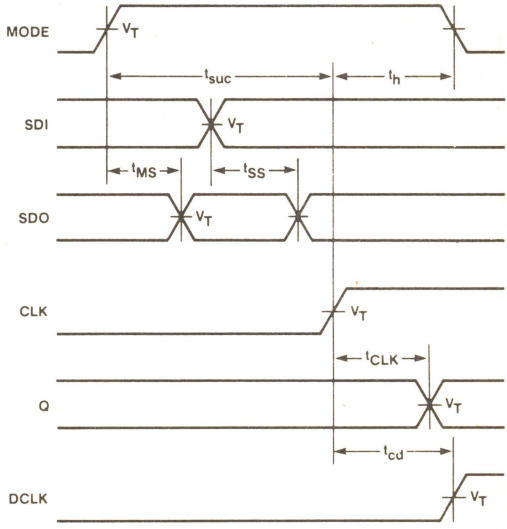
In a system environment, Diagnostics-On-Chip allows complete controllability and observability of registers buried within devices or within blocks of logic on the printed circuit board. Access is permitted through a scan path, requiring only one input to access a number of internal nodes. The scan path within each DOC chip can be connected to any other DOC chip by connecting SDO to SDI in series, and applying DCLK and MODE signals in parallel to all DOC chips. This allows a single scan path to be routed throughout a board, increasing the testability of the board and system.

## Diagnostics Switching Waveforms

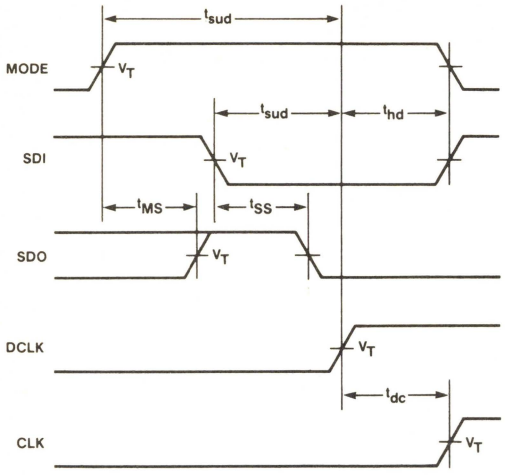



Note: Sequencer operates independently when MODE is low

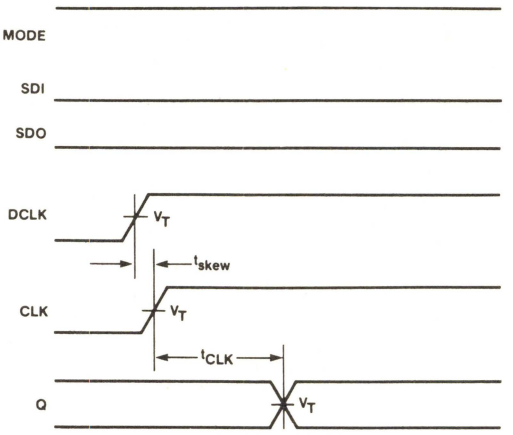
**Shifting Diagnostic Register**



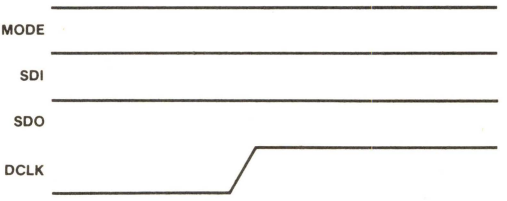

**Load Output Register from Diagnostic Register**




**Load Diagnostic Register from Output Register**




**Swap Registers**




**No Operation**

Notes:
1. $V_T$ = 1.5 V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

## Diagnostics Operating Conditions

| SYMBOL | PARAMETER | | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | STD | | | A | | | |
| | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{wd}$ | Width of DCLK | Low | 30 | 15 | | 30 | 15 | | ns |
| | | High | 30 | 15 | | 30 | 15 | | |
| $t_{suc}$ | Setup time from MODE to CLK | | 30 | 15 | | 30 | 15 | | ns |
| $t_{sud}$ | Setup time from MODE or SDI to DCLK | | 30 | 15 | | 30 | 15 | | ns |
| $t_h$ | Hold time for MODE to CLK | | 15 | 10 | | 15 | 10 | | ns |
| $t_{hd}$ | Hold time for MODE or SDI to DCLK | | 15 | 10 | | 15 | 10 | | ns |
| $t_{dc}$ | DCLK to CLK separation if not simultaneous | MODE = H | 50 | 30 | | 50 | 30 | | ns |
| $t_{cd}$ | CLK to DCLK separation if not simultaneous | MODE = H | 40 | 20 | | 40 | 20 | | ns |
| $t_{skew}$ | CLK to DCLK (DCLK to CLK) separation in swap mode | MODE = H | | 10 | 5 | | 10 | 5 | ns |

## Diagnostics Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | COMMERCIAL | | | | | | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | STD | | | A | | | |
| | | | | MIN | TYP | MAX | MIN | TYP | MAX | |
| $t_{DS}$ | DCLK to SDO | MODE = L | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | 20 | 35 | | 20 | 35 | ns |
| $t_{SS}$ | SDI to SDO | MODE = H | | | 10 | 25 | | 10 | 25 | ns |
| $t_{MS}$ | MODE to SDO | | | | 10 | 25 | | 10 | 25 | ns |
| $f_{MAXd}$ | Maximum diagnostic shift frequency ($1/(t_{wdh} + t_{wdl})$) | | | 16.6 | 33 | | 16.6 | 33 | | MHz |

5

## Design Example

As an introduction to designing with the PMS14R21, an example application will be used to demonstrate the state machine design file required for PALASM 2 software. PALASM 2 software can verify the design and then create a file for programming the device.

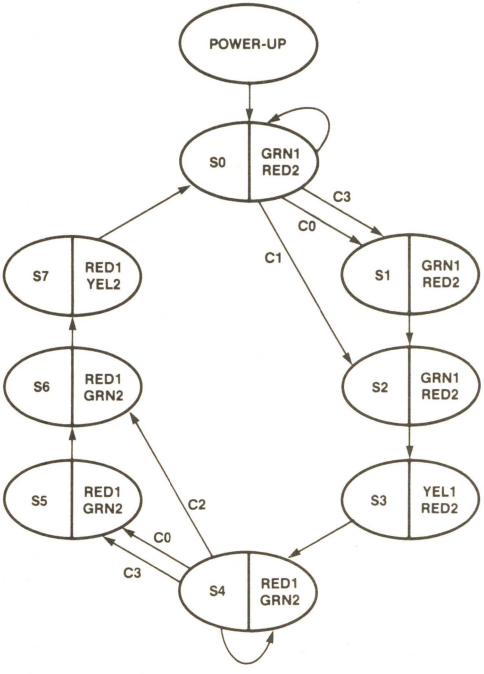
The example is a traffic signal controller for an intersection of two one-way streets. The inputs are active-high sensors for each direction and a clock, and the outputs drive the six active-high lights for the two signals.

The state diagram for this application is shown below. Each bubble represents a single state, with the state name on the left and the state outputs on the right. Each arrow represents a potential transition between states, with the conditions required for that transition (if any) next to the arrow.

Translation of this design to a PALASM 2 software file is straightforward. The following sections will be discussed in detail:

- Declaration and pin definitions (equivalent to other PALASM 2 software files)
- Other definitions required for state machines and the PMS14R21
- Transitions and their conditions
- State outputs
- Definitions of conditions
- Optional simulation section.

The complete design file is shown on the next page.




Traffic Controller State Diagram

## DECLARATION Section

This section documents the design. The CHIP statement consists of the design name, the part number (PMS14R21), and the pin list for pins 1-24 of the device.

## STATE Section

The STATE section begins with the keyword STATE and has three parts: a block of general definitions in the design, the transition equations, and the state outputs.

## General Definitions

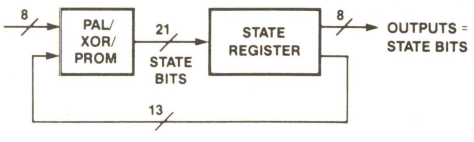
The first general definition specifies one of two types of state machine. Either MOORE_MACHINE or MEALY_MACHINE may be entered, according to the design. The difference between the two types is shown below. In this case, the outputs are a function of only the state, so MOORE_MACHINE is specified.

The next general definition specifies the programmable function pin. Preset or Enable are selected by specifying MASTER_RESET or OUTPUT_ENABLE. MASTER_RESET is used, although neither is functionally required in this design.

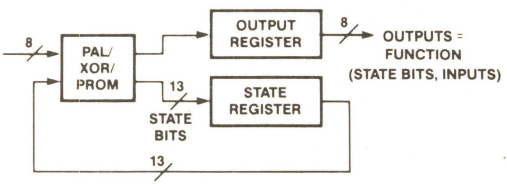
The next general definitions describe defaults for outputs not otherwise defined for a transition. First is an optional OUTPUT_HOLD statement that lists all outputs that are to hold value as a default. It is not used in this design.

Second is the optional DEFAULT_OUTPUT statement that lists default output values. High, Low, and Don't Care are selected with <pin name>, /<pin name>, and %<pin name>, respectively. In the example, the signal lights default to off, or Low.

Next, the optional DEFAULT_BRANCH statement specifies the next state when not otherwise specified. The state may be a state name, or can be HOLD_STATE or NEXT_STATE to hold or go to the next state in the design file. It is not needed in this design.




Moore Machine




Mealy Machine

```
TITLE       Traffic Controller              ;Description Section
PATTERN     State Machine
REVISION    1
AUTHOR      Jane Engineer
COMPANY     Monolithic Memories
DATE        January 30, 1987

CHIP        S_MACHINE PMS14R21
            CLK DCLK SEN1 SEN2 I2 I3 I4 I5 I6 I7 SDI GND
            PRESET SDO RED1 YEL1 GRN1 RED2 YEL2 GRN2 O1 O0 MODE VCC

;This example demonstrates the conversion of a state diagram to a
;PALASM 2 input file, or PDS file.  The example is a traffic signal
;controller for an intersection of two one-way streets.  The inputs
;are active high sensors for each direction and a clock, and the
;outputs drive the six active-high lights for the two signals.

STATE                       ;Specifies state machine format

;General Definitions
   MOORE_MACHINE            ;Outputs function of state only
   MASTER_RESET             ;Programmable pin selects preset

   DEFAULT_OUTPUT /RED1 /YEL1 /GRN1 /RED2 /YEL2 /GRN2
                            ;Outputs default to Low, or off

;Transition Equations
   POWER_UP:= VCC -> S0 ;Goto State0 after power-up (or preset)
   S0:= C3 -> S1        ;From State0: if C3 true, goto State1
      + C0 -> S1        ;              or if C0 true, goto State1
      + C1 -> S2        ;              or if C1 true, goto State2
          +-> S0        ;             Otherwise goto State0 (hold)
   S1:=VCC -> S2        ;From State1: goto State2 unconditionally
   S2:=VCC -> S3        ;From State2: goto State3 unconditionally
   S3:=VCC -> S4        ;From State3: goto State4 unconditionally
   S4:= C3 -> S5        ;From State4: if C3 true, goto State5
      + C0 -> S5        ;              or if C0 true, goto State5
      + C2 -> S6        ;              or if C2 true, goto State6
          +-> S4        ;             Otherwise goto State4 (hold)
   S5:=VCC -> S6        ;From State5: goto State6 unconditionally
   S6:=VCC -> S7        ;From State6: goto State7 unconditionally
   S7:=VCC -> S0        ;From State7: goto State0 unconditionally

;Output Equations
   S0.OUTF := GRN1*RED2    ;In State0 outputs GRN1 and RED2 are High
   S1.OUTF := GRN1*RED2    ;In State1 outputs GRN1 and RED2 are High
   S2.OUTF := GRN1*RED2    ;In State2 outputs GRN1 and RED2 are High
   S3.OUTF := YEL1*RED2    ;In State3 outputs YEL1 and RED2 are High
   S4.OUTF := RED1*GRN2    ;In State4 outputs RED1 and GRN2 are High
   S5.OUTF := RED1*GRN2    ;In State5 outputs RED1 and GRN2 are High
   S6.OUTF := RED1*GRN2    ;In State6 outputs RED1 and GRN2 are High
   S7.OUTF := RED1*YEL2    ;In State7 outputs RED1 and YEL2 are High

 CONDITIONS
    C0 = /SEN1*/SEN2     ;C0 is true when SEN1 and SEN2 Low
    C1 = /SEN1* SEN2     ;C1 is true when SEN1 Low and SEN2 High
    C2 =  SEN1*/SEN2     ;C2 is true when SEN1 High and SEN2 Low
    C3 =  SEN1* SEN2     ;C3 is true when SEN1 and SEN2 High

 ;(SIMULATION is optional - not shown here)
```

5

## Transition Equations

The next section of the file is the transition equations. These are directly equivalent to the arrows in the state diagram, using the state names and the conditions defined later in the file. For example, the second state equation defines the four transition arrows from state S0:

```
S0: = C3 -> S1
    + C0 -> S1
    + C1 -> S2
      +-> S0
```

which means: when in state S0, if condition C3 is true go to state S1, or if condition C0 is true go to state S1, or if condition C1 is true go to state S2, otherwise go to state S0 (hold). The last line has no condition, so that if none of the preceding conditions is true, this transition will occur by default. Note that conditions and states cannot be grouped by parentheses.

The first transition equation in the example specifies the transition from the power-up state. The condition is VCC because the transition always occurs on the clock signal. If a Mealy machine were being designed, the power-up state would also require definition of the outputs for transition to the next state. Note that the power-up state is also entered when the preset pin is asserted.

## Output Equations

The next section of the design file lists the output equations, which specify the output values for each state. The first output equation, S0.OUTF := GRN1*RED2, means that in state S0, GRN1 and RED2 will be High (lights GRN1 and RED2 will be on) while the other outputs assume their default values (Low, or off). Note that for a Mealy machine, equations specify the outputs on transitions to next states, and can include conditions.

## CONDITIONS Section

The next section of the design file defines the conditions used in the STATE section. In this case, C0 = /SEN1*/SEN2 means that condition C0 is true when the inputs SEN1 and SEN2 are Low (sensors are not activated). Conditions equal a sum of products of the device inputs.

## SIMULATION Section

The SIMULATION section provides a means of verifying the design. State machine simulation is similar to Boolean equation simulation, with a few added features. This section is optional.

## Other Design Considerations

PALASM 2 software automatically assigns states to PROM addresses, and automatically arranges the product terms and XOR feedback signals. The automatic assignment optimizes the use of both the PAL and PROM arrays for the most efficient use of resources.

Transition equations are limited to four-way branches, but up to sixteen-way branching can be performed in two clock cycles. This requires the addition of a temporary state after the first four-way branch.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

# Programmable Logic Sequencer
# PLS105-37 PLS167-33 PLS168-33

## ADVANCE INFORMATION

## Features/Benefits

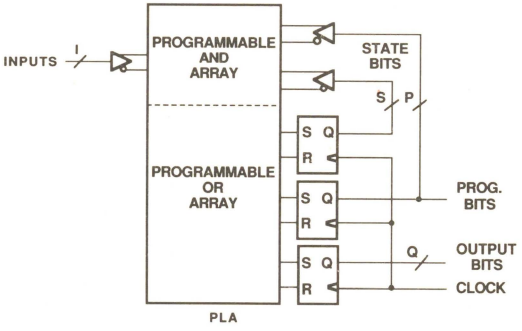
- Field-programmable replacement for sequential control logic
- Advanced Mealy state machine/sequencer architecture
- Programmable AND/programmable OR array for flexibility
- Fabricated with high performance oxide-isolated process
  - $f_{MAX}$ of 37 MHz at 180 mA max for PLS105
  - $f_{MAX}$ of 33 MHz at 180 mA max for PLS167, PLS168
- Full drive: 24 mA IOL, three-state outputs
- Six buried state registers
- Dedicated hardware features to enhance testability
  - Diagnostic mode access to buried state register
  - Register preload and power-up preset of all flip-flops
  - On-chip test array for 100% AC testing
- User-programmable multifunction pin for asynchronous flip-flop preset/output enable
- Automatic "Hold" state via positive edge-triggered clocked S-R flip-flops
- Security fuse hides proprietary designs from competitors
- Integrated software and hardware development system, supported by current PALASM® 2 software and standard PAL® device programmers
- Available in 24- and 28-pin SKINNYDIP® (plastic or ceramic) and 28-pin PLCC packages

| PART NO. | PINS | INPUTS | FLIP-FLOPS | OUTPUTS |
|---|---|---|---|---|
| PLS105-37 | 28 | 16 | 14 | 8 |
| PLS167-33 | 24 | 14 | 12 | 6 |
| PLS168-33 | 24 | 12 | 14 | 8 |

## Application Areas

- Microcoded numeric processors
- Graphics and image processors
- Data storage peripherals - disk and magtape
- Bus arbitration and handshake generation
- Protocol conversion and interface
- Address generation for memory access
- General control functions including:
  - Counters
  - Shift registers
  - Timing waveform generators
  - Mealy state machines

## Generic Sequencer Structure




## Description

The PLS family is a field-programmable replacement for sequential logic. It functions as a Mealy state machine with a registered output. The PLS family utilizes the familiar AND/OR PLA logic structure to implement sum-of-product equations. Both arrays are user-programmable to implement transition terms causing changes in the internal state register or output register.

The device is fabricated in Monolithic Memories' high-speed, oxide-isolated bipolar process; it offers significant speed improvement (fMAX of 37 MHz) over competing parts while preserving their low-power operation.

The PLS family is fully supported by industry standard CAD tools, including PALASM 2 software and ABEL™ software from Data I/O and CUPL™ software from Personal CAD Systems. Device programming is accomplished by using standard PAL device programmers.

5

PAL®, PALASM® and SKINNYDIP® are registered trademarks of Monolithic Memories Inc.
ABEL™ is a trademark of Data I/O Corp.
CUPL™ is a trademark of Personal CAD Systems.

10235A
JANUARY 1988

## Feature Description

State machines contain conditional input logic, state memory and output generation logic. The PLS family is built around a programmable AND/OR logic array which serves as both conditional input and output generation logic. Forty-eight product or transition terms are found in the AND array. It is driven from several sources: sixteen external inputs, ten internal feedbacks from the state register, and the complement term.

The internal variable (C) is known as the complement array, and directly implements the "else" logic clause at any state. All transition terms can include True, False, or Don't Care states of the controlling variables. The OR array merges one or more product terms to generate the desired user logic functions for the output and next-state registers.

The state and output registers are both implemented with edge-triggered, clocked S-R flip-flops. If neither input is active, the flip-flop will retain its contents when clocked. This free "hold" state saves product terms. The registers may change only on the low-to-high transition of the clock pulse.

Starting the state machine in a known state is facilitated by circuitry which unconditionally loads a "1" into each flip-flop during power-up or by using the asynchronous Preset function. The Preset input can be converted to a three-state Output Enable function, as an additional user-programmable function.

Testing of the PLS family is enhanced by the dedicated hardware features of Diagnostic Mode, register preload and a test array. The Diagnostic Mode provides visibility of the internal state register when strobed by a super-voltage control signal. Register preload allows use of an arbitrary test vector set to fully exercise all state transitions and guard against unreachable states and deadlock loops. The internal test array is used to fully guarantee all AC specifications and timing margins.

A security fuse in the PLS family is used to hide proprietary designs from examination by competitors. This last fuse, when programmed, blocks external access and reading of the fuse array and internal device configuration.

## Definition of Signals

$I_{0\text{-}15}$ Inputs to AND array

$Q_{0\text{-}7}$ Output register outputs

CLK Output/state register clock

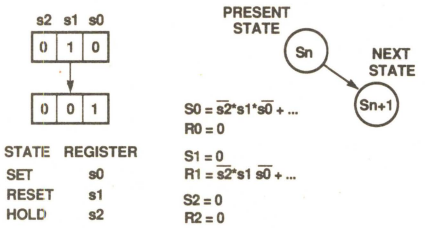
$P/\overline{E}$ Programmable asynchronous function pin; default is active high Preset (all registers go high), programmed is active low Output Enable

$S_{0\text{-}5}$ Internal state register bits used with feedback to AND array

$P_{0\text{-}3}$ Multiple-use pins–user selectable as either state register (used with feedback) or output register functions–present on PLS167 and PLS168 only

## Logic Function

TYPICAL STATE TRANSITION:

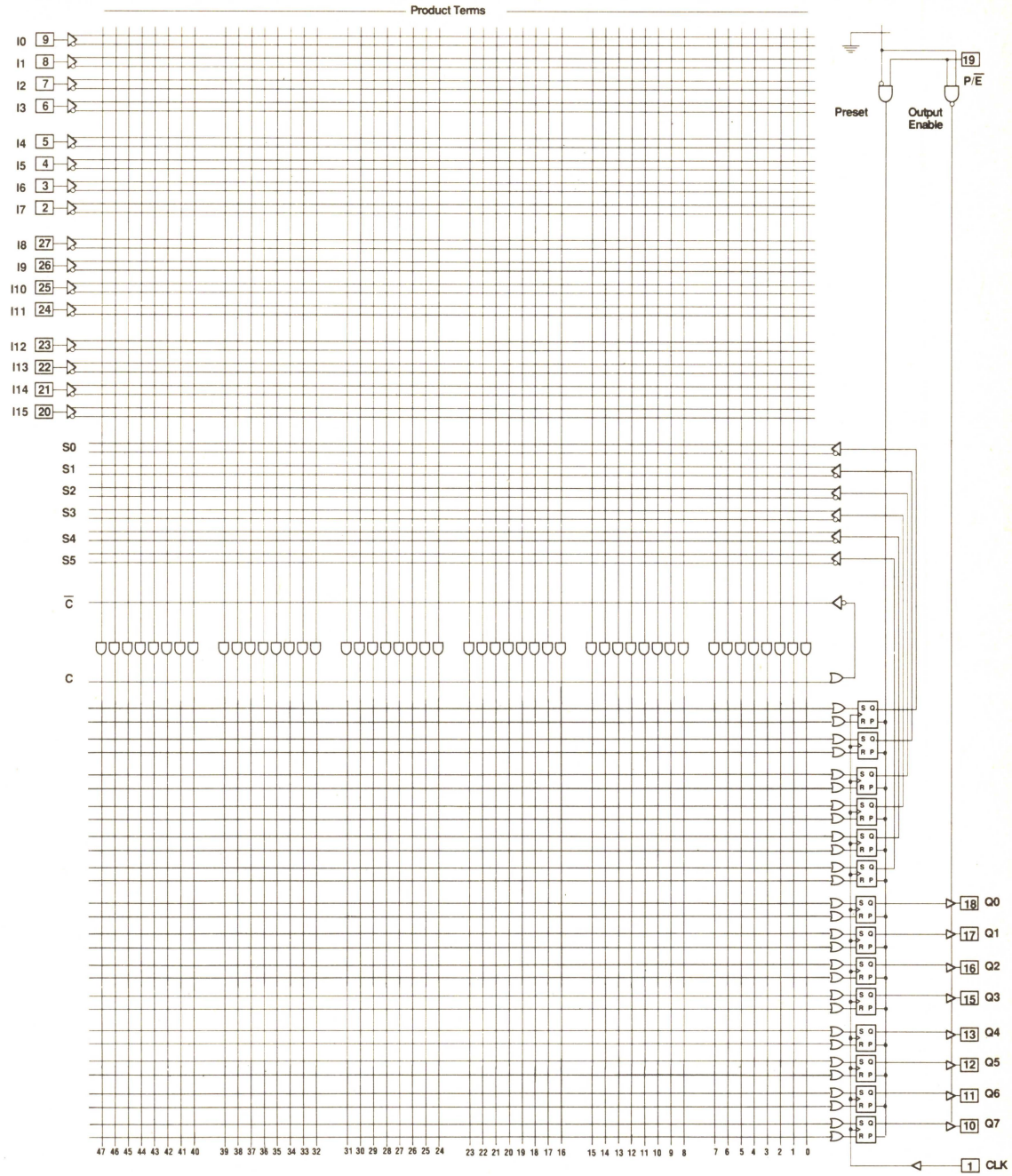



## Typical Operation

The details of device operation may be illustrated by the simple state transition indicated. The state register initially contains 010 and will become 001 after the next clock. In order for this to occur, state bit 0 must be set, state bit 1 must be reset and state bit 2 must hold its value. The transition term fragment listed produces this result. The S0 and R1 product terms detect the bit pattern for the current state (010) and produce a logic one. All other terms evaluate to a zero, producing the transition to state 001.

## Logic Diagram

PLS105-37



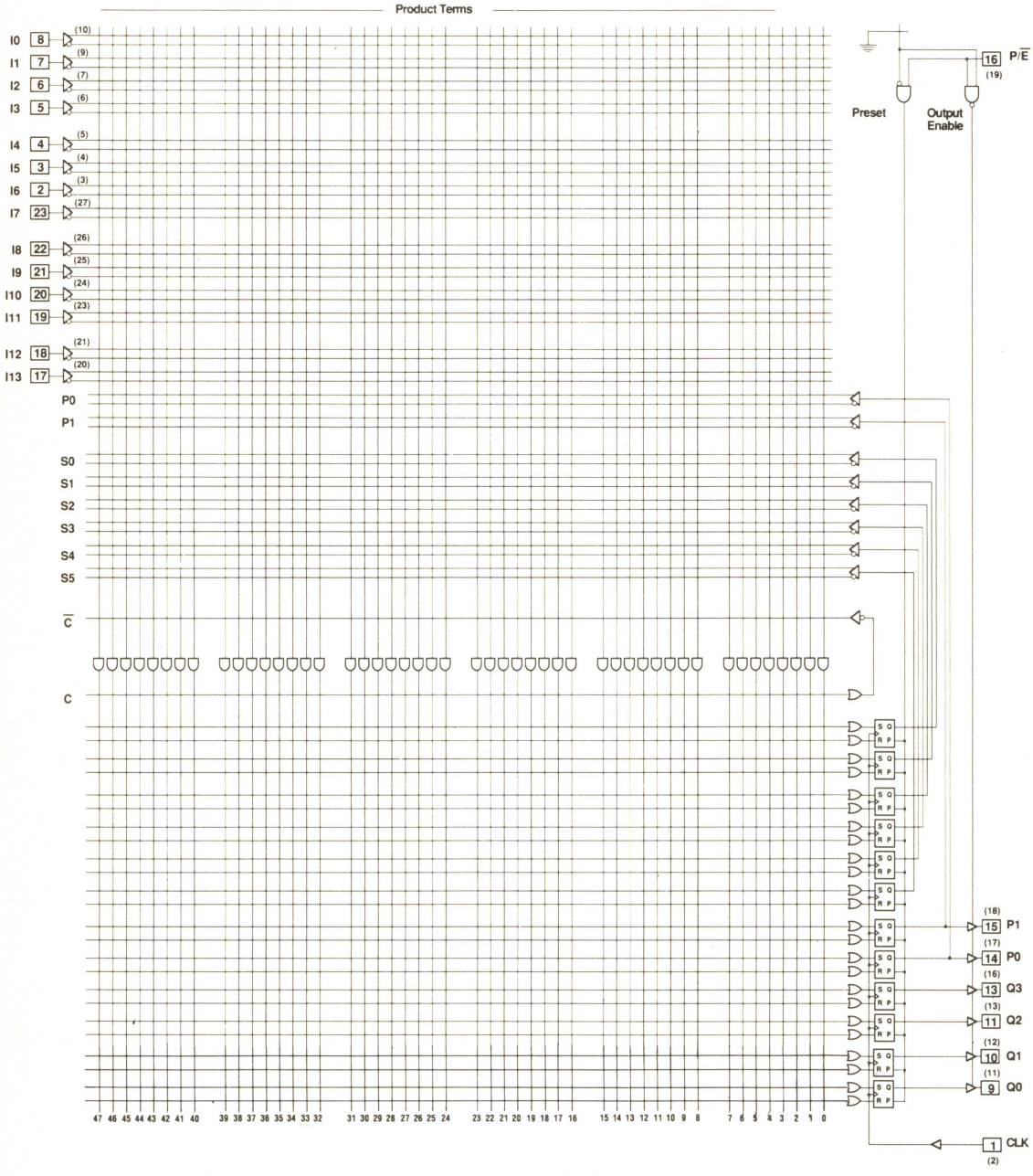

Notes:

1. All AND gate inputs with a programmed link float to a logic "1."
2. All OR gate inputs with a programmed link float to a logic "0."

## Logic Diagram

PLS167-33

## DIP (PLCC) Pinout



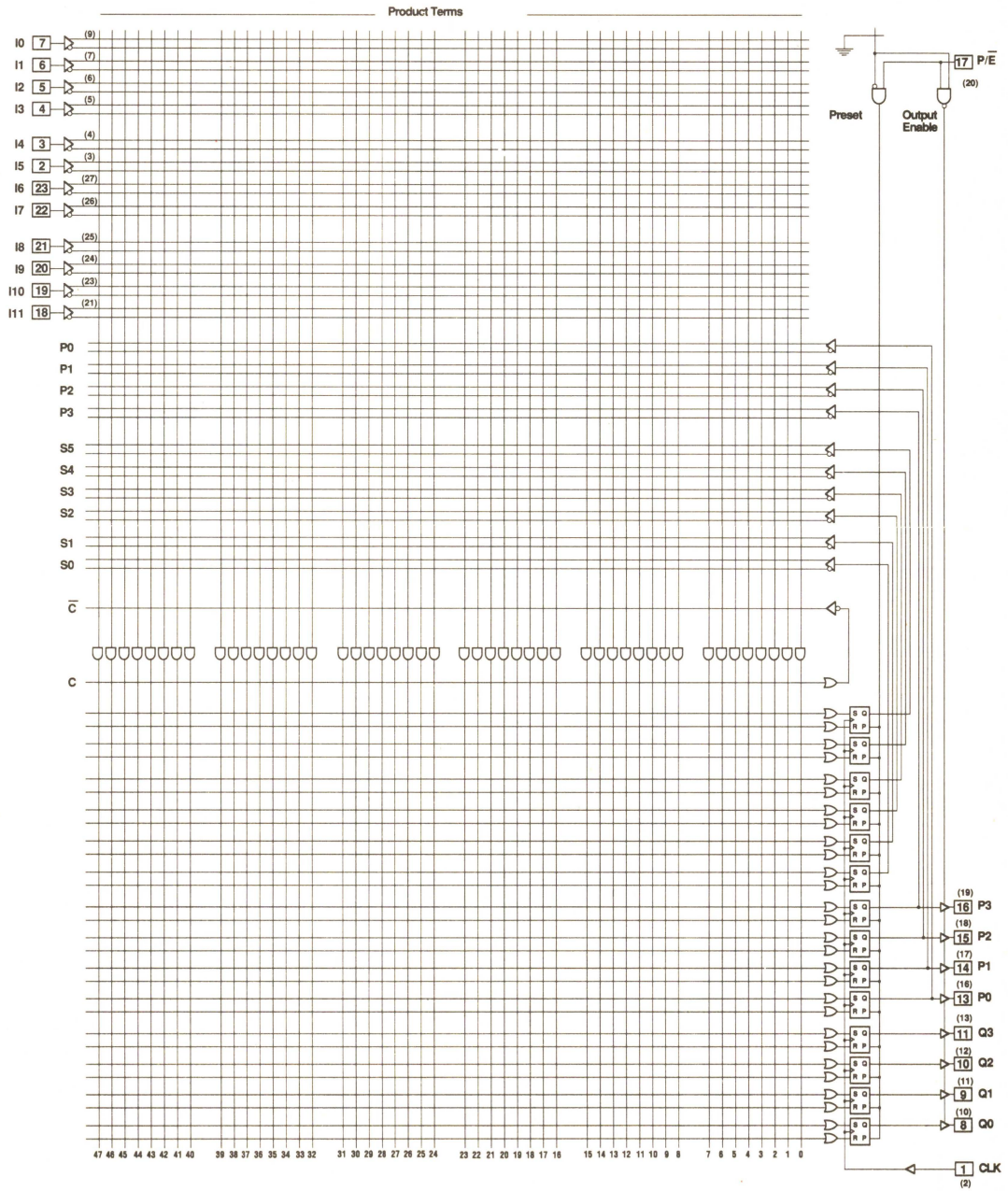

Notes:

1. All AND gate inputs with a programmed link float to a logic "1."
2. All OR gate inputs with a programmed link float to a logic "0."

## Logic Diagram

PLS168-33

## DIP (PLCC) Pinout

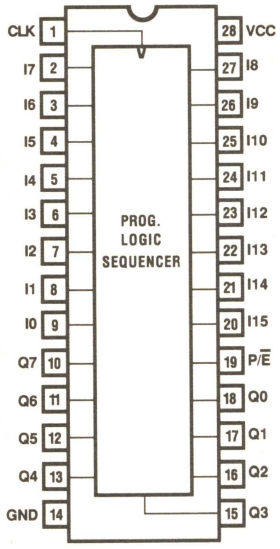



Notes:

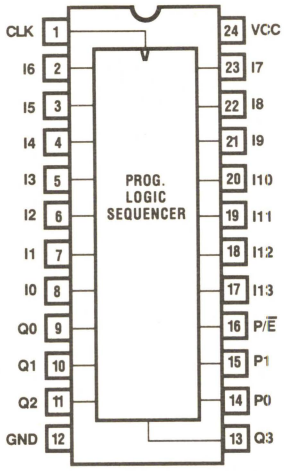
1. All AND gate inputs with a programmed link float to a logic "1."
2. All OR gate inputs with a programmed link float to a logic "0."

## Pin Configurations

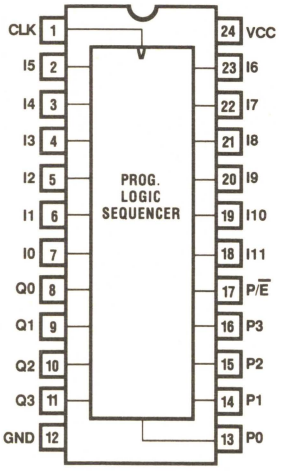
PLS105-37
28 Pin DIP



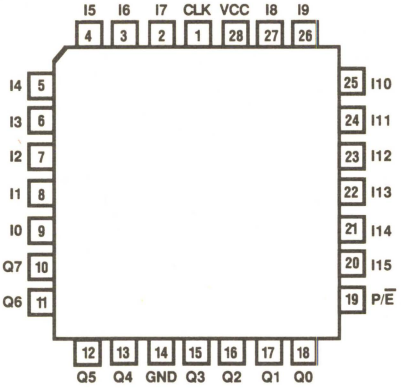

PLS167-33
24 Pin DIP



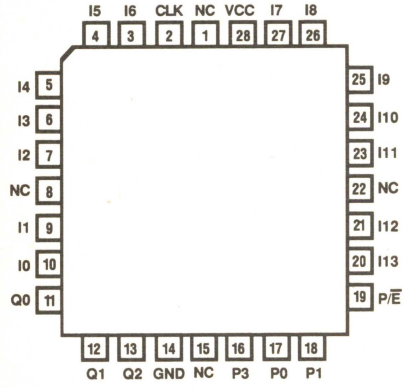

PLS168-33
24 Pin DIP



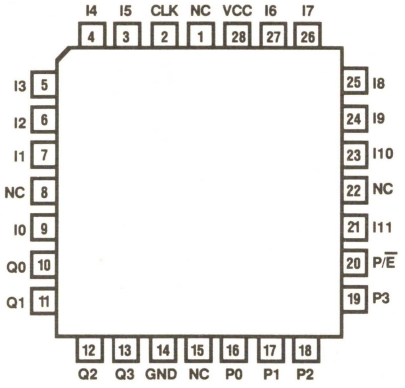

PLS105-37
28 Pin PLCC

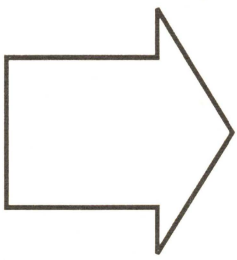



PLS167-33
28 Pin PLCC




PLS168-33
28 Pin PLCC





## Data Sheets

- TTL/CMOS PAL Devices
- TTL/CMOS AmPAL Devices
- PROSE/PLS Sequencers
- FPC/PEG Sequencers
- ECL PAL Devices
- HAL/ZHAL Devices
- Military PAL Devices
- Logic Cell Array
- Electrical Definitions


IMPORTANT
NOTES

# Am29PL141

Fuse Programmable Controller (FPC)

## Distinctive Characteristics

- Implements complex fuse programmable state machines
- 7 conditional inputs, 16 outputs
- 64 words of 32-bit-wide microprogram memory
- Serial Shadow Register (SSR™) diagnostics on chip (programmable option)
- 29 high-level microinstructions
  - Conditional branching
  - Conditional looping
  - Conditional subroutine call
  - Multiway branch
- 20 MHz clock rate, 28-pin DIP

## General Description

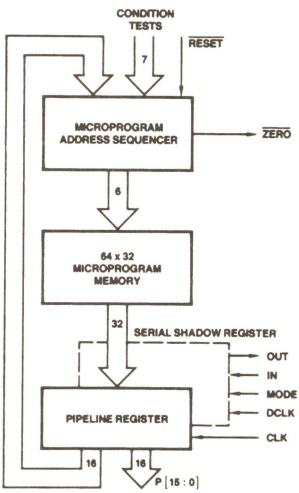
The Am29PL141 is a single-chip Fuse Programmable Controller (FPC) which allows implementation of complex state machines and controllers by programming the appropriate sequence of microinstructions. A repertoire of jumps, loops, and subroutine calls, which can be conditionally executed based on the test inputs, provides the designer with powerful control flow primitives.

The Am29PL141 FPC also allows distribution of intelligent control throughout the system. It off-loads the central controller by distributing FPCs as the control for various self-contained functional units, such as register file/ALU, I/O, interrupt, diagnostic, and bus control units.

A microprogram address sequencer is the heart of the FPC. It provides the microprogram address to an internal 64-word by 32-bit PROM. The fuse programming algorithm is almost identical to that used for AMD's Programmable Array Logic family.

As an option, the Am29PL141 may be programmed to have on chip SSR diagnostics capability. Microinstructions can be serially shifted in, executed, and the results shifted out to facilitate system diagnostics.

## Block Diagram




BDR02340

## Related Products

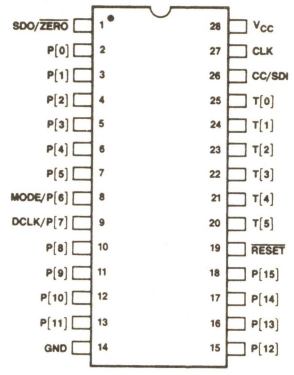
| Part No. | Description |
|---|---|
| Am2914 | Vectored Priority Interrupt Controller |
| Am29100 | Controller Family Products |

SSR is a trademark of Advanced Micro Devices, Inc.

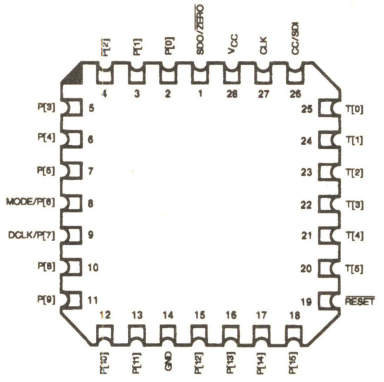
04179F/0
JANUARY 1988

## Connection Diagrams

### Top View



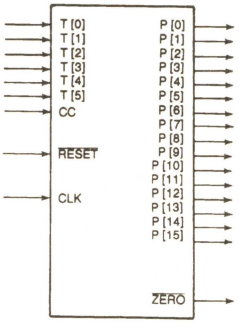

CDR04480




CD009110

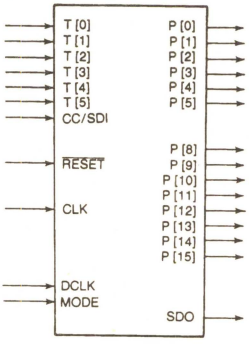
Note: Pin 1 is marked for orientation.

## Logic Symbols




LS002131

**Normal Configuration**



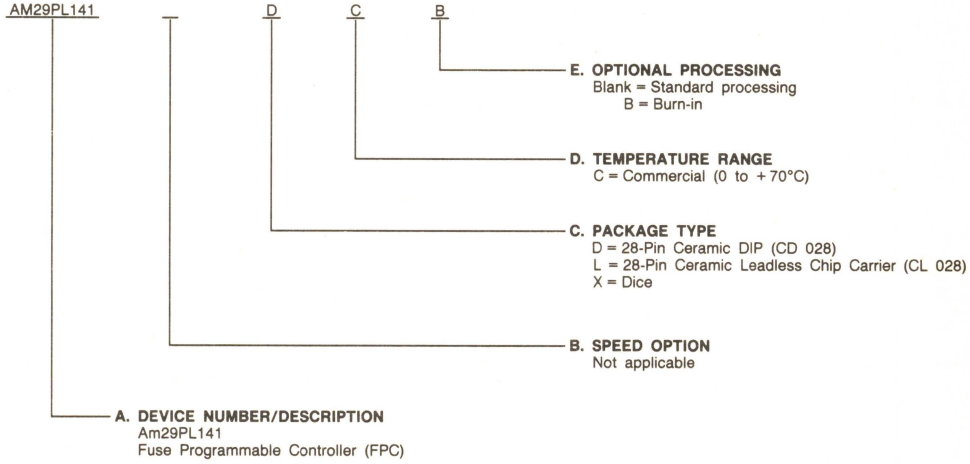

LS002140

**SSR™ Diagnostics Configuration**

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AM29PL141 | DC, DCB, LC, XC |

**Valid Combinations**

Valid Combinations list cnfigurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released valid combinations, and to obtain additional data on AMD's standard military grade products.

5

## Ordering Information

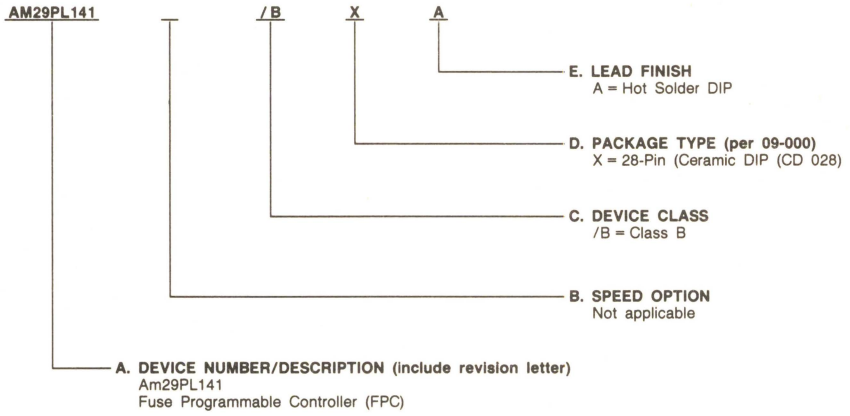
### APL and CPL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. CPL (Controlled Products List) products are processed in accordance with MIL-STD-883C, but are inherently non-compliant because of package, solderability, or surface treatment exceptions to those specifications. The order number (Valid Combination) is formed by a combination of:

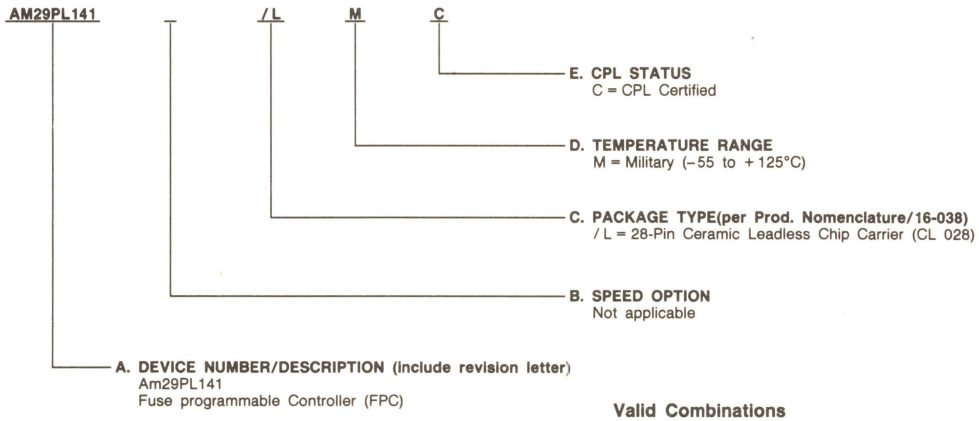
**APL Products:**
**A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**

**CPL Products:**
**A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. CPL Status**

### APL Products




### CPL Products




| Valid Combinations | | |
|---|---|---|
| APL | Am29PL141 | /BXA |
| CPL | Am29PL141 | /LMC |

### Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

### Group A Tests

Group A tests consists of Subgroups:
1, 2, 3, 7, 8, 9, 10, 11

## Pin Description

**CC[SDI] Condition Code ((TEST) Input)**
When the TEST (P[24:22]) field of the executing microinstruction is set to 6 (binary 110), CC is selected to be the conditional input. (Note: In SSR diagnostic configuration, CC is also the Serial Data Input SDI.)

**CLK Clock (Input)**
The rising edge clocks the microprogram counter, count register, subroutine register, pipeline register, and EQ flag.

**P[15:8] (Outputs)**
Upper eight, general-purpose microprogram control outputs. They are enabled by the OE signal from the microprogram pipeline register. When OE is HIGH, P[15:8] are enabled, and when LOW, P[15:8] are three-stated.

**P[7:0] [DLCK, MODE] (Outputs)**
Lower
Lower eight, general-purpose microprogram control outputs. They are permanently enabled. (Note: in the SSR diagnostic configuration, P[7] becomes the diagnostic clock input DCLK and P[6] becomes the diagnostic control input MODE.)

**$\overline{\text{RESET}}$**
Synchronous reset input. When it is low, the output of the PC MUX is forced to the uppermost microprogram address (63). On the next rising clock edge, this address (63) is loaded into the microprogram counter, the microinstruction at location 63 is loaded into the pipeline register and the EQ flag is cleared. The CREG and SREG values are indeterminate on reset.

**T[5:0]**
Test inputs. In conditional microinstructions, the inputs can be used as individual condition codes selected by the TEST field in the pipeline register. The T[5:0] inputs can also be used as a branch address when performing a microprogram branch, or as a count value.

**$\overline{\text{ZERO}}$ [SDO]**
Zero output. A Low state indicates that the CREG value is zero. (Note: In the SSR diagnostic configuration, $\overline{\text{ZERO}}$ becomes the Serial Data output SDO. This change is only on the output pin; internally, the zero detect functions is unchanged.)

## Functional Description

Figure 1, the block diagram of the Am29PL141 FPC, shows logic blocks and interconnecting buses. These allow parallel performance of different operations in a single microinstruction. The FPC consists of four main logic blocks: the microprogram memory, microaddress control logic, condition code selection logic, and microinstruction decode. A fifth optional block is the Serial Shadow Register (SSR).

The microprogram memory contains the user-defined instruction flow and output sequence. The microaddress control logic addresses the microprogram memory. This control logic supports high-level microinstruction functions including conditional branches, subroutine calls and returns, loops, and multiway branches. The condition code selection logic selects the condition code input to be tested when a conditional microinstruction is executed. The polarity of the selected condition code input is controlled by the POL bit in the microword. The microinstruction decode generates the control signals necessary to perform the microinstruction specified by the microinstruction part (P[31 : 16]) of the microword. The SSR enables in-system testing that allows isolation of problems down to the IC level.

## Microprogram Memory

The FPC microprogram memory is a 64-word by 32-bit PROM with a 32-bit pipeline register at its output. The upper 16 bits (P[31 : 16]) of the pipeline register stay internal to the FPC and form the microinstruction to control address sequencing. The format for microinstructions is: a one-bit synchronous Output Enable OE, a five-bit OPCODE, a one-bit test polarity select POL, a three-bit TEST condition select field, and a six-bit immediate DATA field. The DATA field is used to provide branch addresses, test input masks, and counter values.

The lower 16 bits (P[15 : 0]) of the pipeline register are brought out as user-defined, general purpose control outputs. The upper eight control outputs (P[15 : 8]) are three-stated when OE is programmed as a LOW. The lower eight control bits (P[7 : 0]) are always enabled.

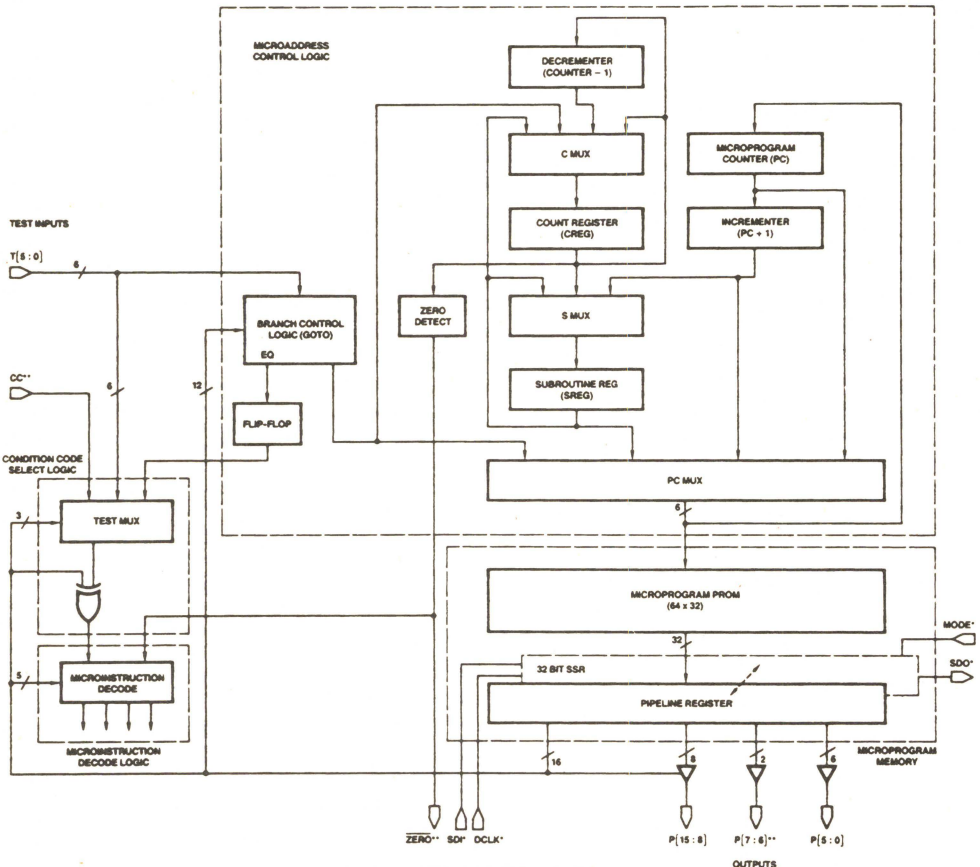



**Figure 1. Am29PL141 Block Diagram**

BDR02330

*Note: These pins available only in SSR mode.
**Note: These pins available only in normal mode.

## Microaddress Control Logic

The microaddress control logic consists of five smaller logic blocks. These are:

PC MUX – The microprogram counter multiplexer

P CNTR – Microprogram counter (PC) and incrementer (PC + 1)

SUBREG – Subroutine register (SREG) with subroutine mux (S MUX)

CNTR – Count register (CREG) with counter mux (C MUX), decrementer (COUNTER–1) and zero detect

GOTO – Specialized branch control logic

The PC MUX is a six-bit, four-to-one multiplexer. It selects either the PC, PC+1, SREG, or GOTO output as the next microaddress input to the microprogram memory and to the PC. The PC thus always contains the address of the microinstruction in the pipeline register. During a Reset, the PC MUX output is forced to all ones, selecting location 63 of the microprogram memory.

The P CNTR block consists of a six-bit register (PC) driving a six-bit combinatorial incrementer (PC+1). Either the present or the incremented values of PC can address the microprogram PROM. The incremented value of PC can be saved as a subroutine return address. The present PC value can address the microprogram PROM when waiting for a condition to become valid. PC+1 addresses the microprogram PROM for sequential microprogram flow, for unconditional microinstructions, and as a default for conditional microinstructions.

The SUBREG block consists of a six-bit, three-to-one multiplexer (S MUX) driving a six-bit register (SREG). The three possible SREG inputs are PC+1, CREG, and SREG. SREG normally operates as a one-deep stack to save subroutine return addresses. PC+1 is the input source when performing subroutine calls and PC MUX is the output destination when performing return from subroutine.

The CNTR block consists of a six-bit, four-to-one multiplexer (C MUX); driving a six-bit register (CREG); a six-bit, combinatorial decrementer (COUNTER-1); and a zero detection circuit. The CNTR logic block is typically used for timing functions and iterative loop counting.

The SUBREG and CNTR can be considered as one logic block because of their unique interaction. To explain this interaction, notice that both have an additional input source and output destination not used in typical operation—each other. This allows the CREG to be an additional stack location when not used for counting, and the SREG to be a nested count location when not used as a stack location. Thus, the SREG and CREG can operate in three different modes:

1. As a separate one-deep stack and counter.
2. As a two-deep stack.
3. As a two-deep nested counter.

The GOTO logic block serves three functions:

1. It provides a six-bit count value from the DATA Field in the pipeline register (P[21 : 16]) or from the TEST inputs (T[5 : 0]) masked by the DATA Field (P[21 : 16]). (This is represented by T*M.)
2. It provides a branch address from the DATA Field in the pipeline register (P[21 : 16]) or from the TEST inputs (T[5 : 0]) masked by the DATA Field (P[21 : 16]). (This is represented by T*M.)
3. It compares the TEST inputs (T[5 : 0]) masked by the DATA Field (P[21 : 16]), called T*M, to the CONSTANT Field from the pipeline register (P[27 : 22]). If a match occurs, the EQ Flip-flop is set. EQ remains unchanged if there is no match. Constant field bits that correspond to marked test bits must be ZERO.

The EQ flag can be tested by the condition code selection logic. Multiple tests of any group of T inputs in a manner analogous to Sum-of-Products can be performed since a no match comparison does not reset the EQ flag. Any conditional branch on EQ will reset the EQ flag. Conditional returns on EQ will not change the EQ flag. $\overline{\text{RESET}}$ input LOW will reset the EQ flag.

NOTE: A zero in the DATA Field blocks the corresponding bit in the TEST Field; a one activates the corresponding bit.

The constant field bits that correspond to masked test field bits must be zero. A zero is substituted for masked test field bits. The 'POL' bit is a "don't care" when using test inputs to load registers.

## Condition Code Selection Logic

The condition code selection logic consists of an eight-to-one multiplexer. The eight test conditon inputs are the device inputs (CC and T[5 : 0]) and the EQ flag. The TEST field P[24 : 22] selects one of the eight conditions to test.

The polarity bit POL in the microinstructions allows the user to test for either a true or false conditon. Refer to Table 2 for details.

## Microinstruction Decode

The microinstruction decoder is a PLA that generates the control for 29 different microinstructions. The decoder's inputs include the OPCODE Field (P[30 : 26]), the zero detection output from the CNTR, and the selected test condition code from the conditional code selection logic.

## Am29PL141 SSR Diagnostics Option

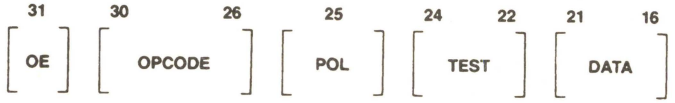
As a programmable option, the Am29PL141 FPC may be configured to contain Serial Shadow Register (SSR) diagnostics capability. SSR diagnostics is a simple, straightforward method of in-system testing that allows isolation of problems down to the IC level.

The SSR diagnostics configuration activates a 32-bit-wide, D-type register, called a "shadow" register, on the pipeline register inputs. The shadow register can be serially loaded from the SDI pin, parallel loaded from the pipeline register, or held. The pipeline register can be loaded from the microprogram memory in normal operation or from the shadow register during diagnostics. A redefinition of four device pins is required to control the different diagnostics functions. CC also functions as the Serial Data Input (SDI), $\overline{\text{ZERO}}$ becomes the Serial Data Output (SDO), P[7] becomes the diagnostic clock (DCLK), and P[6] becomes the diagnostic mode control (MODE). The various diagnostic and normal modes are shown in Table 1.

Serially loading a test microinstruction into the shadow register and parallel loading the shadow register contents into the pipeline register forces execution of the test microinstruction. The result of the test microinstruction can then be clocked into the pipeline register, as in normal operation mode, parallel loaded into the shadow register, and serially shifted out for system diagnostics.

The general microinstruction format is shown below:

**Am29PL141 General Microinstruction Format**




DFR00730

WHERE:

OE = Synchronous Output Enable for P[15:8].

OPCODE = A five-bit opcode field for selecting one of the twenty-eight single data field microinstructions.

POL = A one-bit test condition polarity select.
0 = Test for true (HIGH) condition.
1 = Test for false (LOW) condition.

TEST = A three-bit test condition select.

| TEST[2:0] | UNDER TEST |
|---|---|
| 000 | T[0] |
| 001 | T[1] |
| 010 | T[2] |
| 011 | T[3] |
| 100 | T[4] |
| 101 | T[5] |
| 110 | CC |
| 111 | EQ |

DATA = A six-bit conditional branch microaddress, test input mask, or counter value field designated as PI in microinstruction mnemonics.

The special two data field comparison microinstruction format is shown below:

**Am29PL141 Comparison Microinstruction Format**




DFR00740

WHERE:

OE = Synchronous Output Enable for P[15:8].

OPCODE = Compare microinstruction (binary 100).

CONSTANT = A six-bit constant for equal to comparison with T*M.

DATA = A six-bit mask field for masking the incoming T[5:0] inputs.

## Table 1.

| Inputs | | | | Outputs | | | Operation |
|---|---|---|---|---|---|---|---|
| SDI | MODE | DCLK | CLK | SDO | Shadow Register | Pipeline Register | |
| D | L | ↑ | H,L,↑ | $S_O$ | $S_{i-1} \leftarrow S_i$<br>$S_{31} \leftarrow D$ | Hold | Serial Right Shift Register |
| X | L | H,L,↓ | ↑ | $S_O$ | Hold | $P_i \leftarrow PROM_i$ | Normal Load Pipeline Register from PROM |
| L | H | ↑ | H,L,↓ | L | $S_I \leftarrow P_i$ | Hold | Load Shadow Register from Pipeline* Register |
| X | H | H,L,↓ | ↑ | SDI | Hold | $P_I \leftarrow S_i$ | Load Pipeline Register from Shadow Register |
| H | H | ↑ | H,L,↓ | H | Hold | Hold | Hold Shadow Register |

*S7, S6 are undefined. $S_{15} - S_8$ load from the source driving pins P[15] – P[8]. If P[31] in the microword is a ONE, $S_{15}$-$S_8$ are loaded from the pipeline register. If P[31] in the microword is a ZERO, $S_{15} - S_8$ are loaded from an external source.

### Function Table Definitions

**Inputs**

H = HIGH X = Don't Care
L = LOW ↑ = LOW-to-HIGH transition
↓ = High-to-Low transition

## Table 2.

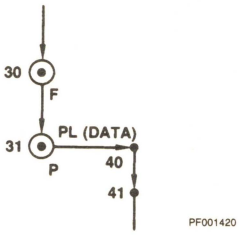
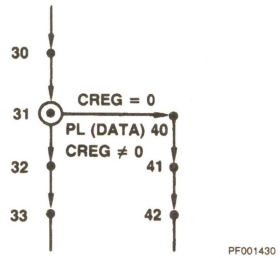
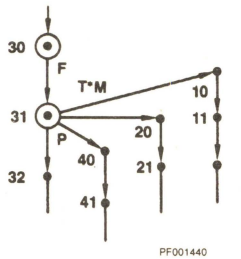
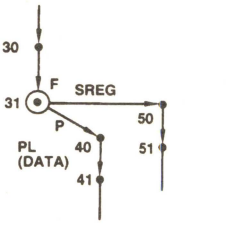
| Input Condition Being Tested | POL | Condition |
|---|---|---|
| 0 | 0 | Fail |
| 0 | 1 | Pass |
| 1 | 0 | Pass |
| 1 | 1 | Fail |

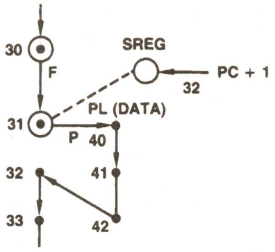
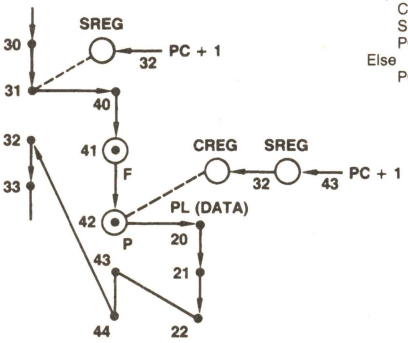
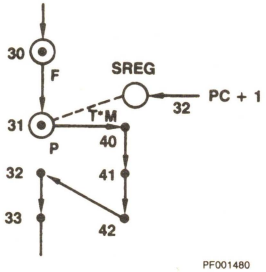
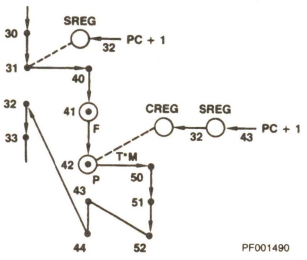
5

## Am29PL141 Microinstruction Set Definition

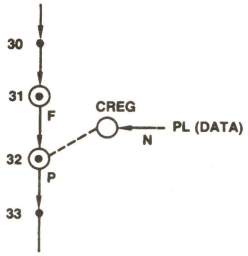
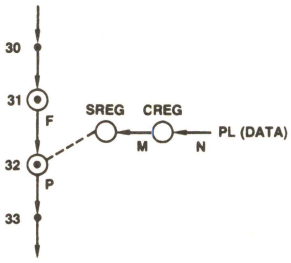
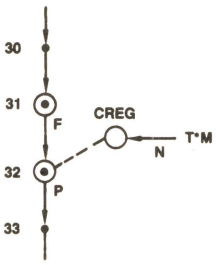
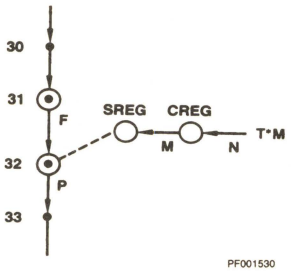
● = Other instruction
⊙ = Instruction being described
$\epsilon$ = Register in part

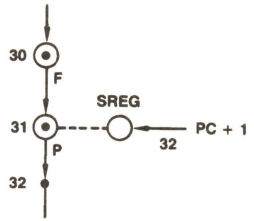
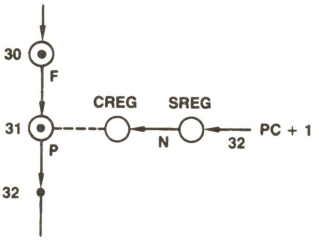
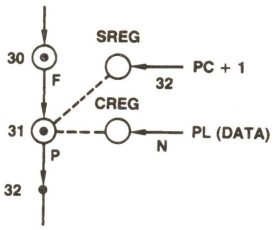
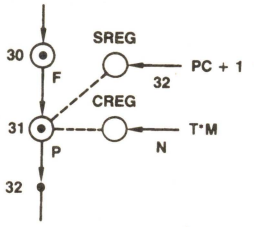
P = Test Pass
F = Test Fail
M,N are arbitrary values in the CREG or SREG

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 19 | GOTOPL | **If (cond) Then Go To Pipeline**<br>Conditional branch to the address in the PL (DATA field). The EQ flag will be reset if the test field selects it and the condition passes. |  <br>PF001420 | If ( cond = true ) Then<br>PC = PL(data)<br>Else<br>PC = PC + 1 |
| 0B | GOTOPLZ | **If (CREG = O) Then Go To Pipeline**<br>Conditional branch, when the CREG is equal to zero, to the address in the PL (DATA field). This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and the CREG is equal to zero. |  <br>PF001430 | If ( CREG = 0 ) Then<br>PC = PL(data)<br>Else<br>PC = PC + 1 |
| 0F | GOTOTM | **If (cond) Then Go To TM**<br>Conditional branch to the address defined by the T*M (T[5:0] under bitwise mask from the DATA field). This microinstruction is intended for multiway branches. The EQ flag will be reset if the test field selects it and the condition passes. |  <br>PF001440 | If ( cond = true ) Then<br>PC = T*M<br>Else<br>PC = PC + 1 |
| 18 | FORK | **If (cond) Then Go To Pipeline Else Go To (SREG)**<br>Conditional branch to the address in the PL (DATA field) or the SREG. A branch to PL is taken if the condition is true and a branch to SREG if false. The EQ flag will be reset if the test field selects it and the condition passes. |  <br>PF001451 | If ( cond = true ) Then<br>PC = PL(data)<br>Else<br>PC = SREG |

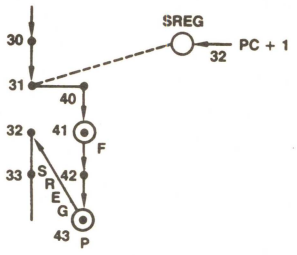
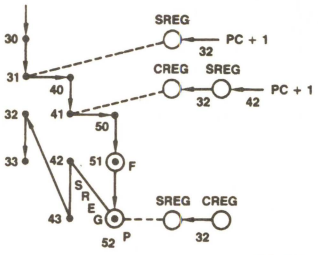
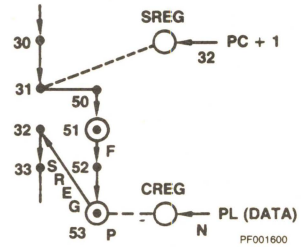
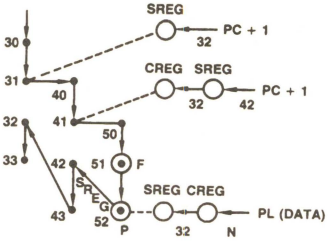
| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 1C | CALPL | **If (cond) Then Call Pipeline** Conditional jump to subroutine at the address in the PL (DATA field). The PC + 1 is pushed into the SREG as the return address. The EQ flag will be reset if the test field selects it and the condition passes. |   PF001461 | If ( cond = true ) Then<br>SREG = PC + 1<br>PC = PL(data)<br>Else<br>PC = PC + 1 |
| 1D | CALPLN | **If (cond) Then Call Pipeline, Nested** Conditional jump to subroutine at the address in the PL (DATA field) nested. The SREG and CREG are treated as a two-deep stack, the PC + 1 is pushed into the SREG as the return address and the previous SREG value is transferred into the CREG as a nested return address. The EQ flag will be reset if the test field selects it and the condition passes. |   PF001470 | If ( cond = true ) Then<br>CREG = SREG<br>SREG = PC + 1<br>PC = PL(data)<br>Else<br>PC = PC + 1 |
| 1E | CALTM | **If (cond) Then Call TM** Conditional jump to subroutine at the address specified by the T*M (T[5:0] under bitwise mask from the DATA field). The PC + 1 is pushed into the SREG as the return address. The EQ flag will be reset if the test field selects it and the condition passes. |   PF001480 | If ( cond = true ) Then<br>SREG = PC + 1<br>PC = T*M<br>Else<br>PC = PC + 1 |
| 1F | CALTMN | **If (cond) Then Call TM, Nested** Conditional jump to subroutine at the address specified by the T*M (T[5:0] under bitwise mask from the DATA field) nested. The PC + 1 is pushed into the SREG as the return address and the previous SREG value is transferred into the CREG as a nested return address. The EQ flag will be reset if the test field selects it and the condition passes. |   PF001490 | If ( cond = true ) Then<br>CREG = SREG<br>SREG = PC + 1<br>PC = T*M<br>Else<br>PC = PC + 1 |

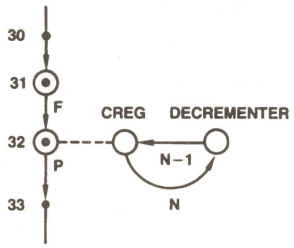
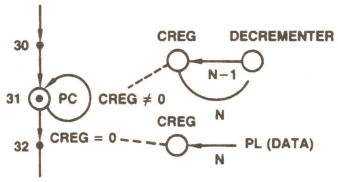
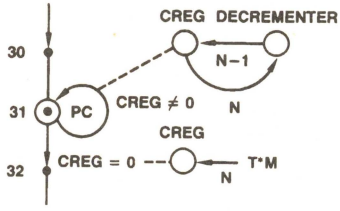
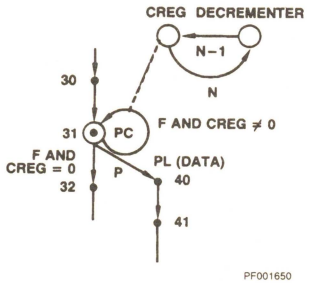
5

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 04 | LDPL | **If (cond) Then Load Pipeline**<br>Conditional Load the CREG from the PL (DATA field). |  <br>PF001510 | If ( cond = true ) Then<br>CREG = PL(data)<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 05 | LDPLN | **If (cond) Then Load Pipeline, Nested**<br>Conditional load the CREG from the PL (DATA field) nested. The CREG and SREG are treated as a two-deep nested count register, the previous CREG value is pushed into the SREG as a nested count, and the CREG is loaded from PL. |  <br>PF001500 | If ( cond = true ) Then<br>SREG = CREG<br>CREG = PL(data)<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 06 | LDTM | **If (cond) Then Load TM**<br>Conditional load the CREG from the T*M (T[5:0] inputs under bitwise mask from the DATA field). |  <br>PF001520 | If ( cond = true ) Then<br>CREG = T*M<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 07 | LDTMN | **If (cond) Then Load TM, Nested**<br>Conditional load the CREG from the T*M (T[5:0] inputs under bitwise mask from the DATA field) nested. The SREG and CREG are treated as a two-deep nested count register, the previous CREG value is transferred into the SREG and the CREG is loaded from T*M. |  <br>PF001530 | If ( cond = true ) Then<br>SREG = CREG<br>CREG = T*M<br>PC = PC + 1<br>Else<br>PC = PC + 1 |

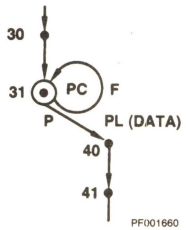
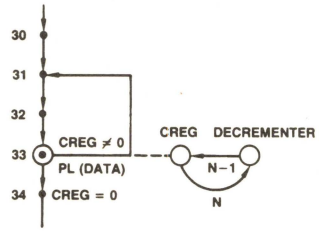
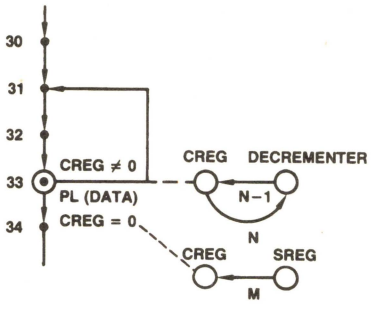
| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 15 | PSH | **If (cond) Then Push**<br>Conditional push the PC + 1 into the SREG. |  <br>PF0001540 | If ( cond = true ) Then<br>SREG = PC + 1<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 17 | PSHN | **If (cond) Then Push, Nested**<br>Conditional push the PC + 1 into the SREG nested. This microinstruction treats the SREG and CREG as a two-deep stack, PC + 1 is pushed into SREG and the previous value in SREG is transferred into the CREG. |  <br>PF001550 | If ( cond = true ) Then<br>CREG = SREG<br>SREG = PC + 1<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 14 | PSHPL | **If (cond) Then Push, Load Pipeline**<br>Conditional push the PC + 1 into the SREG and load the CREG from the PL (DATA field). |  <br>PF001560 | If ( cond = true ) Then<br>CREG = PL(data)<br>SREG = PC + 1<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 16 | PSHTM | **If (cond) Then Push, Load TM**<br>Conditional push the PC + 1 into the SREG and load the CREG from the T*M (T[5:0] under bitwise mask from the DATA field). |  <br>PF001570 | If ( cond = true ) Then<br>CREG = T*M<br>SREG = PC + 1<br>PC = PC + 1<br>Else<br>PC = PC + 1 |

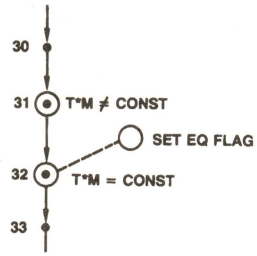
5

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 02 | RET | **If (cond) Then Return**<br>Conditional return from subroutine. The SREG provides the return from subroutine address. |  <br>PF001580 | If ( cond = true ) Then<br>PC = SREG<br>Else<br>PC = PC + 1 |
| 03 | RETN | **If (cond) Then Return Nested**<br>Conditional return from nested subroutine. This microinstruction treats the SREG and CREG as a two-deep stack providing the SREG value as a return address and the CREG value as a nested return address that is transferred into the SREG. |  <br>PF001590 | If ( cond = true ) Then<br>PC = SREG<br>SREG = CREG<br>Else<br>PC = PC + 1 |
| 00 | RETPL | **If (cond) Then Return, Load Pipeline**<br>Conditional return from subroutine and load the CREG from the PL (DATA field). The SREG provides the return from subroutine address. |  <br>PF001600 | If ( cond = true ) Then<br>CREG = PL(data)<br>PC = SREG<br>Else<br>PC = PC + 1 |
| 01 | RETPLN | **If (cond) Then Return Nested, Load Pipeline**<br>Conditional return from nested subroutine and load the CREG from the PL (DATA field). This microinstruction treats the SREG and CREG as a two-deep stack providing the SREG value as a return address and the CREG value as a nested return address that is transferred into the SREG. |  <br>PF001610 | If ( cond = true ) Then<br>PC = SREG<br>SREG = CREG<br>CREG = PL(data)<br>Else<br>PC = PC + 1 |

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 09 | DEC | **If (cond) Then Decrement**<br>Conditional decrement of the CREG. |  <br>PF001620 | If ( cond = true ) Then<br>CREG = CREG – 1<br>PC = PC + 1<br>Else<br>PC = PC + 1 |
| 0C | DECPL | **While (CREG ≠ 0) Wait Else Load Pipeline**<br>Conditional Hold until the counter is equal to zero, then load CREG from the PL (DATA field). This microinstruction is intended for timing waveform generation. If the CREG is not equal to zero, the same microinstruction is refetched while CREG is decremented. Timing is complete when the CREG is equal to zero, causing the next microinstruction to be fetched and the CREG to be reloaded from PL. This instruction does not depend on the pass/fail condition. |  <br>PF001630 | While ( CREG < > 0)<br>CREG = CREG – 1<br>PC = PC<br>End While<br>CREG = PL(data)<br>PC = PC + 1 |
| 0E | DECTM | **While (CREG ≠ 0) Wait Else Load TM**<br>Conditional Hold until the counter is equal to zero, then load CREG from the T*M (T[5:0] under bitwise mask from the DATA field). This microinstruction is intended for timing waveform generation. If the CREG is not equal to zero, the same microinstruction is refletched while the CREG is decremented. Timing is complete when the CREG is equal to zero, causing the next microinstruction to be fetched and the CREG to be reloaded from T*M. This instruction does not depend on the pass/fail condition. |  <br>PF001640 | While ( CREG < > 0 )<br>CREG = CREG – 1<br>PC = PC<br>End While<br>CREG = T*M<br>PC = PC + 1 |
| 1B | DECGOPL | **If (cond) Then Go To Pipeline Else While (CREG ≠ 0) Wait**<br>Conditional Hold/Count. The current microinstruction will be refetched and the CREG decremented until the condition under test becomes true or the counter is equal to zero. If the condition becomes true, a branch to the address in the PL (DATA field) is executed. If the counter becomes zero without the condition becoming true, a CONTINUE is executed. The EQ flag will be reset if the test field selects it and the condition passes. |  <br>PF001650 | While ( cond = false )<br>If ( CREG < > 0 )<br>CREG = CREG – 1<br>PC = PC<br>Else<br>PC = PC + 1<br>End While<br>PC = PL(data) |

5

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 1A | WAIT | **If (cond) Then Go To Pipeline Else Wait**<br>Conditional Hold. The current microinstruction will be refetched and executed until the condition under test becomes true. When true, a branch to the address in the PL (DATA field) is executed. The EQ flag will be reset if the test field selects it and the condition passes. |  <br>PF001660 | If ( cond = true ) Then<br>PC = PL(data)<br>Else<br>PC= PC |
| 08 | LPPL | **While (CREG ≠ 0) Loop to Pipeline**<br>Conditional loop to the address in the PL (DATA field). This microinstruction is intended to be placed at the bottom of an iterative loop. If the CREG is not equal to zero, it is decremented (signifying completion of an iteration), and a branch to the PL address (top of the loop) is executed. If the CREG is equal to zero, looping is complete and the next sequential microinstruction is executed. This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and CREG is not equal to zero. |  <br>PF001670 | While ( CREG < > 0)<br>CREG = CREG – 1<br>PC = PL (data)<br>End While<br>PC = PC + 1 |
| 0A | LPPLN | **While (CREG ≠ 0) Loop to Pipeline Else Nest**<br>Conditional loop to the address in the PL (DATA field) nested. The SREG and CREG are treated as a two-deep nested count register, and the microinstruction is intended to be placed at the bottom of an "inner-nested" iterative loop. If the CREG is not equal to zero, the CREG is decremented (signifying completion of an iteration), and a branch to the PL address (top of the loop) is executed. If the CREG is equal to zero, the inner loop is complete, and the count value for the outer loop is transferred from the SREG into the CREG. This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and CREG is not equal to zero. |  <br>PF001680 | While ( CREG < > 0)<br>CREG = CREG – 1<br>PC = PL(data)<br>End While<br>CREG = SREG<br>PC = PC + 1 |

| Opcode | Mnemonics | Description | Execution Example | Register Transfer Description |
|---|---|---|---|---|
| 0D | CONT | **Continue** The next sequential microinstruction is fetched unconditionally. |  PF001690 | PC = PC + 1 |
| 10 – 13 (100XX binary) | CMP | **Compare TM to Pipeline (DATA)** This microinstruction performs bitwise exclusive-or of T*M (T[5:0] under bitwise mask from the DATA field) with CONSTANT (P[27:22]). If T*M equals CONSTANT, the EQ flag is set to one which may be branched on in a following microinstruction. If not equal, the EQ flag is unaffected. This allows sequences of compares, in a manner analogous to sum-of-products, to be performed which can be followed by a single conditional branch if one or more of the comparisons were true. Note: The EQ flag is set to zero on reset or when EQ is selected as the condition in a branch. Conditional returns on EQ leave the flag unchanged. **Constant field bits that correspond to masked test field bits must be zero.** This instruction does not depend on the pass/fail condition. |  PF001701 | Compare T*M and PL(data)<br>EQ = ((T [5:0] .AND. DATA).<br>XNOR. CONSTANT) .OR. EQ |

5

## Microinstruction Set Table

| Code | Mnemonics | Definition | CREG Content | Pass | | | | Fail | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | PC MUX | SREG | CREG | EQ | PC MUX | SREG | CREG | EQ |
| 00 | RETPL | Return: Load Pipeline | X | SREG | Hold | Data | NC | PC + 1 | Hold | Hold | NC |
| 01 | RETPLN | Return Nested: Load Pipeline | X | SREG | CREG | Data | NC | PC + 1 | Hold | Hold | NC |
| 02 | RET | Return | X | SREG | Hold | Hold | NC | PC + 1 | Hold | Hold | NC |
| 03 | RETN | Return Nested | X | SREG | CREG | Hold | NC | PC + 1 | Hold | Hold | NC |
| 04 | LDPL | Load Pipeline | X | PC + 1 | Hold | Data | NC | PC + 1 | Hold | Hold | NC |
| 05 | LDPLN | Load Pipeline Nested | X | PC + 1 | CREG | Data | NC | PC + 1 | Hold | Hold | NC |
| 06 | LDTM | Load T*M | X | PC + 1 | Hold | T*M | NC | PC + 1 | Hold | Hold | NC |
| 07 | LDTMN | Load T*M Nested | X | PC + 1 | CREG | T*M | NC | PC + 1 | Hold | Hold | NC |
| 08 | LPPL | Loop Pipeline | ≠ 0 | Data | Hold | DCRMT | Reset | | | | |
| | | | = 0 | PC + 1 | Hold | Hold | NC | | | | |
| 09 | DEC | Decrement | X | PC + 1 | Hold | DCRMT | NC | PC + 1 | Hold | Hold | NC |
| 0A | LPPLN | Loop Pipeline Nested | ≠ 0 | Data | Hold | DCRMT | Reset | | | | |
| | | | = 0 | PC + 1 | Hold | SREG | NC | | | | |
| 0B | GOTOPLZ | Go to Pipeline Zero | ≠ 0 | PC + 1 | Hold | Hold | NC | | | | |
| | | | = 0 | Data | Hold | Hold | Reset | | | | |
| 0C | DECPL | Count/Load Pipeline | ≠ 0 | PC | Hold | DCRMT | NC | | | | |
| | | | = 0 | PC + 1 | Hold | Data | NC | | | | |
| OD | CONT | Continue | X | PC + 1 | Hold | Hold | NC | PC + 1 | Hold | Hold | NC |
| 0E | DECTM | Count/Load T*M | ≠ 0 | PC | Hold | DCRMT | NC | | | | |
| | | | = 0 | PC + 1 | Hold | T*M | NC | | | | |
| 0F | GOTOTM | Go to T*M | X | T*M | Hold | Hold | Reset | PC + 1 | Hold | Hold | NC |
| 10 – 13 (100XX Binary) | CMP | Compare* | X | PC + 1 | Hold | Hold | Set | PC + 1 | Hold | Hold | NC |
| 14 | PSHPL | Push: Load Pipeline | X | PC + 1 | PC + 1 | Data | NC | PC + 1 | Hold | Hold | NC |
| 15 | PSH | Push | X | PC + 1 | PC + 1 | Hold | NC | PC + 1 | Hold | Hold | NC |
| 16 | PSHTM | Push: Load T*M | X | PC + 1 | PC + 1 | T*M | NC | PC + 1 | Hold | Hold | NC |
| 17 | PSHN | Push Nested | X | PC + 1 | PC + 1 | SREG | NC | PC + 1 | Hold | Hold | NC |
| 18 | FORK | Fork | X | Data | Hold | Hold | Reset | SREG | Hold | Hold | NC |
| 19 | GOTOPL | Go to Pipeline | X | Data | Hold | Hold | Reset | PC + 1 | Hold | Hold | NC |
| 1A | WAIT | Hold Pipeline | X | Data | Hold | Hold | Reset | PC | Hold | Hold | NC |
| 1B | DECGOPL | Count: Hold Pipeline | ≠ 0 | Data | Hold | Hold | Reset | PC | Hold | DCRMT | NC |
| | | | = 0 | Data | Hold | Hold | Reset | PC + 1 | Hold | Hold | NC |
| 1C | CALPL | Call Pipeline | X | Data | PC + 1 | Hold | Reset | PC + 1 | Hold | Hold | NC |
| 1D | CALPLN | Call Pipeline Nested | X | Data | PC + 1 | SREG | Reset | PC + 1 | Hold | Hold | NC |
| 1E | CALTM | Call T*M | X | T*M | PC + 1 | Hold | Reset | PC + 1 | Hold | Hold | NC |
| 1F | CALTMN | Call T*M Nested | X | T*M | PC + 1 | SREG | Reset | PC + 1 | Hold | Hold | NC |

EQ = ((T[5:0]. AND. DATA). XNOR. CONSTANT).OR. EQ
CONSTANT field bits that correspond to masked test field bits must be zero.
NC = No Change

**Notes:**
1. (/)Signifies two different operations may occur, depending on the condition.
2. (:) Signifies two parallel operations on the same condition.
3. The EQ flag will be affected only if the test field selects it, with the exception of instructions 10 – 13.

## PROGRAMMING YIELD

AMD programmable logic devices have been designed to insure extremely high programming yields ( > 98%).

AMD programmable logic devices contain many internal test features, including circuitry and extra fuses which allow AMD to test the ability of each part to perform programming before shipping, to assure high programming yields, and correct logical operation for a correctly programmed part. Programming yield losses are most likely due to poor programming socket contact, programming equipment that is out of calibration, or improper usage of said equipment.

## Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)

## Absolute Maximum Ratings

Storage Temperature ..... −65 to +150°C
(Ambient) Temperature Under Bias ..... −55 to +125°C
Supply Voltage to Ground Potential
(Pin 28 to Pin 14) Continuous ..... −0.5 V to +7.0 V
DC Voltage Applied to Outputs
(Except During Programming) ..... −0.5 V to +$V_{CC}$ Max.
DC Voltage Applied to Outputs
During Programming ..... 21 V
DC Output Current, Into Outputs During
Programming (Max Duration of 1 sec) ..... 200 mA
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ..... 0 to +70°C
Supply Voltage ..... +4.75 V to +5.25 V

Military (M) Devices
Temperature ..... −55 to +125°C
Supply Voltage ..... +4.5 V to +5.5 V

*Operating ranges define those limits over which the functionality of the device is guaranteed.*

## DC Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted

| Parameters | Description | Test Conditions | | | Min | Max | Units |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −3.0 mA | COM'L | 2.4 | | Volts |
| | | | $I_{OH}$ = −1.0 mA | MIL | | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 16 mA | COM'L | | 0.50 | Volts |
| | | | $I_{OL}$ = 12 mA | MIL | | | |
| $V_{IH}$ (Note 1) | Input HIGH Level | Guaranteed Input Logical HIGH Voltage for All Inputs | | | 2.0 | | Volts |
| $V_{IL}$ (Note 1) | Input LOW Level | Guaranteed Input Logical LOW Voltage for All Inputs | | | | 0.8 | Volts |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max. $V_{IN}$ = 0.5 V | CLK<br>P [15:6]<br>All other Inputs | | | −1.5<br>−0.55<br>−0.50 | mA |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Max. $V_{IN}$ = 2.4 V | CLK<br>P [15:6]<br>All other Inputs | | | 150<br>100<br>25 | µA |
| $I_I$ | Input HIGH Current | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | | | | 1.0 | mA |
| $I_{SC}$ | Output Short Circuit Current | $V_{CC}$ = Max., $V_{OUT}$ = 0.5 V (Note 2) | | | −20 | −80 | mA |
| $I_{CC}$ | Power Supply Current | $V_{CC}$ = Max. | COM'L | $T_A$ = 0 to 70°C | | 450 | mA |
| | | | | $T_A$ = 70°C | | 400 | |
| | | | MIL | $T_C$ = −55 to 125°C | | 490 | |
| | | | | $T_C$ = 125°C | | 420 | |
| $V_I$ | Input Clamp Voltage | $V_{CC}$ = Min, $I_{IN}$ = −18 mA | | | | −1.2 | Volts |
| $I_{OZH}$ | Output Leakage Current (Note 3) | $V_{CC}$ = MAX, $V_{IL}$ = 0.8 V $V_{IH}$ = 2.0 V | $V_O$ = 2.4 V | | | 100 | µA |
| $I_{OZL}$ | | | $V_O$ = 0.5 V | | | −550 | |

**Notes:**
1. These are absolute values with respect to device ground and all overshoots due to system or tester noise are included.
2. Not more than one output should be tested at a time. Duration of the short circuit should not be more than one second. $V_{OUT}$ = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.
3. I/O pin leakage is the worst case of $I_{OZX}$ or $I_{IX}$ (where X = H or L).

## Switching Characteristics

over operating range unless otherwise specified; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted (APL and CPL products only)

| Parameters | | Description | Test Conditions | COMMERCIAL | | MILITARY | | Units |
|---|---|---|---|---|---|---|---|---|
| | | | | Min. | Max. | Min. | Max. | |
| $t_{PD}$ | 1 | CLK to P[15:0] | See Test Output Load Conditions | | 15 | | 20 | ns |
| | 2 | CLK to $\overline{ZERO}$ | | | 20 | | 25 | ns |
| | 3 | DCLK to SDO | | | 30 | | 35 | ns |
| | 4 | Mode to SDO | | | 30 | | 35 | ns |
| | 5 | SDI to SDO | | | 30 | | 35 | ns |
| $t_S$ | 6 | T[5:0] to CLK (Note 1) | | 40 | | 45 † | | ns |
| | 7 | CC to CLK (Note 1) | | 40 | | 45 † | | ns |
| | 8 | $\overline{RESET}$ to CLK | | 30 | | 35 | | ns |
| | 9 | Mode to CLK | | 30 | | 35 | | ns |
| | 10 | Mode to DCLK | | 30 | | 35 | | ns |
| | 11 | SDI to DCLK | | 30 | | 35 | | ns |
| | 12 | P[15:8] to DCLK | | 30 | | 35 | | ns |
| $t_H$ | 13 | T[5:0] to CLK | | 3 | | 3 | | ns |
| | 14 | CC to CLK | | 3 | | 3 | | ns |
| | 15 | $\overline{RESET}$ to CLK | | 3 | | 3 | | ns |
| | 16 | Mode to CLK | | 3 | | 3 | | ns |
| | 17 | Mode to DCLK | | 3 | | 3 | | ns |
| | 18 | SDI to DCLK | | 3 | | 3 | | ns |
| | 19 | P[15:8] to DCLK | | 3 | | 3 | | ns |
| $t_{PZX}$ | 20 | CLK to P[15:8] Enable | | | 30 | | 35 | ns |
| $t_{PXZ}$ | 21 | CLK to P[15:8] Disable | | | 30 | | 35 | ns |
| $t_{PW}$ | 22 | CLK Pulse Width (HIGH and LOW) | | 20 | | 25 | | ns |
| | 23 | DCLK Pulse Width (HIGH and LOW) | | 30 | | 35 | | ns |
| $t_P$ | 24 | CLK and DCLK Period (Note 1) | | 45 | | 50 † | | ns |

**Notes:**

1. These parameters cannot be measured directly on unprogrammed devices. They are determined as follows:
   a. Measure delay from input (CC, T[5:0], or CLK) to PROM address out in test mode. This will measure the delay through the sequence logic.
   b. Measure setup time from T[5:0] input through PROM test columns to pipeline register in verify test column mode. This will measure the delay through the PROM and register setup.
   c. Measure delay from T[5:0] input to PROM address out in verify test column mode. This will measure the delay through the logic and P[15:0] outputs.

   To calculate the desired parameter measurement the following formula is used:

   Measurement (a) + Measurement (b) – Measurement (c)

   CLK PERIOD:

   CLK (a) + (b) – (c) = CLK PERIOD

   CC to CLK Set-up time:

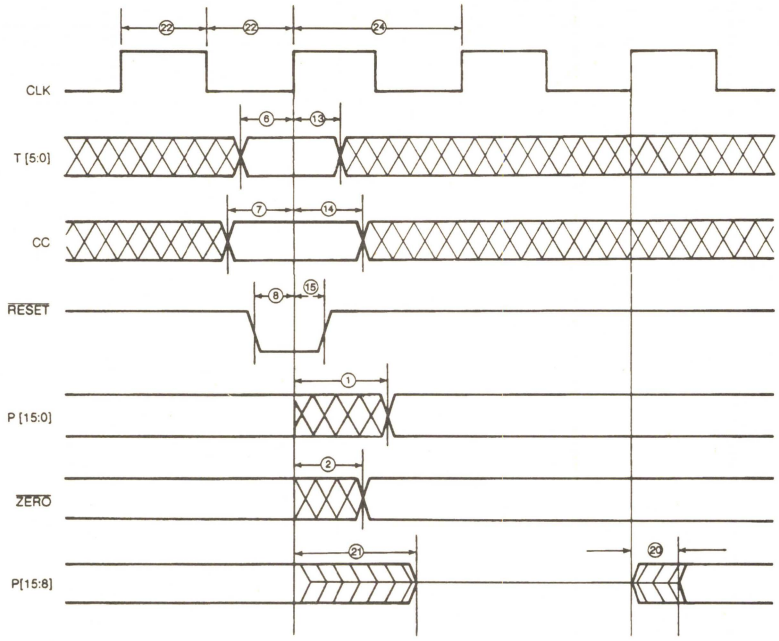
   CC (a) + (b) – (c) = CC to CLK Set-up time

   T[5:0] to CLK Set-up time:

   T[5:0] (a) + (b) – (c) = T[5:0] to CLK Set-up time

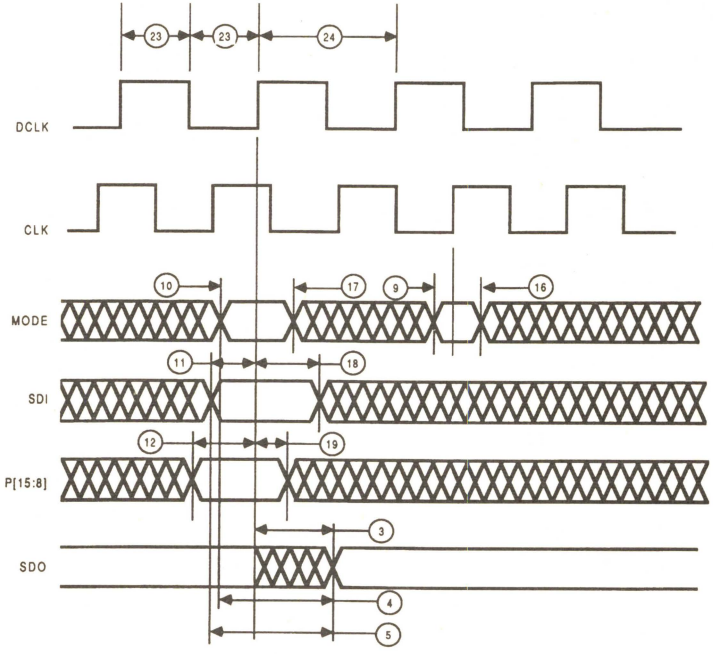
† = Not included in Group A tests

## Switching Waveforms




WF020852

**Normal Configuration**




WF023120

**SSR™ Configuration**

## Test Philosophy and Methods

The following points give the general philosophy that we apply to tests that must be properly engineered if they are to be implemented in an automatic testing environment. The specifics of what philosophies are applied to which test are shown in the data sheet and the data-sheet reconciliation that follow.

### Capacitive Loading for AC Testing

Automatic testers and their associated hardware have stray capacitance that varies from one type of tester to another, but is generally around 50 pF. This, of course, makes it impossible to make direct measurements of parameters that call for smaller capacitive load than the associated stray capacitance. Typical examples of this are the so-called "float delays" that measure the propagation delays in to and out of the high-impedance state and are usually specified at a load capacitance of 5.0 pF. In these cases, the test is performed at the higher load capacitance (typically 50 pF) and engineering correlations based on data taken with a bench setup are used to determine the result at the lower capacitance.

Similarly, a product may be specified at more than one capacitive load. Since the typical automatic tester is not capable of switching loads in mid-test, it is impractical to make measurements at both capacitances even though they may both be greater than the stray capacitance. In these cases, a measurement is made at one of the two capacitances. The result at the other capacitance is determined from engineering correlations based on data taken with a bench setup and the knowledge that certain DC tests are performed in order to facilitate this correlation.

AC loads specified in the data sheet are used for bench testing. Automatic tester loads, which simulate the data-sheet loads, may be used during production testing.

### Threshold Testing

The noise associated with automatic testing, the long inductive cables, and the high gain of bipolar devices frequently give rise to oscillations when testing high-speed circuits. These oscillations are not indicative of a reject device, but instead, of an overtaxed system. To minimize this problem, thresholds are tested at least once for each input pin. Thereafter, "hard" high and low levels are used for other tests. Generally this means that function and AC testing are performed at "hard" input levels.

### AC Testing

AC parameters are specified that cannot be measured accurately on automatic testers because of tester limitations. Data-input hold times fall into this category. In these cases, the parameter in question is tested by correlating the tester to bench data or oscilloscope measurements made on the tester by engineering (supporting data on file).

Certain AC tests are redundant since they can be shown to be predicted by other tests that have already been performed. In these cases, the redundant tests are not performed.

### Output Short-Circuit Current Testing

When performing $I_{OS}$ tests on devices containing RAM or registers, great care must be taken that undershoot caused by grounding the high-state output does not trigger parasitic elements which in turn cause the device to change state. In order to avoid this effect, it is common to make the measurement at a voltage ($V_{output}$) that is slightly above ground. The $V_{CC}$ is raised by the same amount so that the result (as confirmed by Ohm's law and precise bench testing) is identical to the $V_{OUT} = 0$, $V_{CC}$ = Max. case.

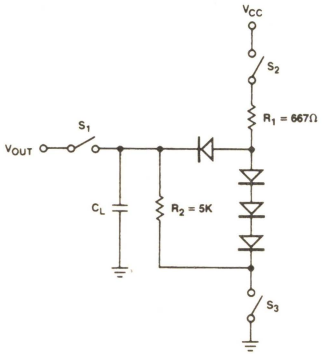
5

## Key to Switching Waveforms

| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| ——— | MUST BE STEADY | WILL BE STEADY |
| ⋱⋱⋱ | MAY CHANGE FROM H TO L | WILL BE CHANGING FROM H TO L |
| ⋰⋰⋰ | MAY CHANGE FROM L TO H | WILL BE CHANGING FROM L TO H |
| ⋈⋈⋈ | DON'T CARE; ANY CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| ⋈⋈⋈ | DOES NOT APPLY | CENTER LINE IS HIGH IMPEDANCE "OFF" STATE |

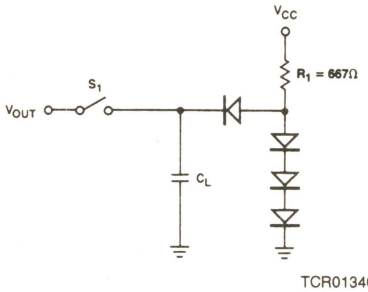
KS000010

## Switching Test Circuits




TCR01330

**A. Three State Outputs**

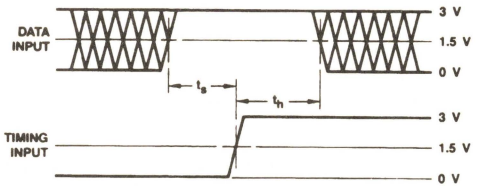



TCR01340

**B. Normal Outputs**

Notes: 1. CL = 50 pF includes scope probe, wiring and stray capacitances without device in test fixture.
2. $S_1$, $S_2$, and $S_3$ are closed during function tests and all AC tests except output enable tests.
3. $S_1$ and $S_3$ are closed while $S_2$ is open for $t_{PZH}$ test.
4. $C_L$ = 5.0 pF for output disable tests.

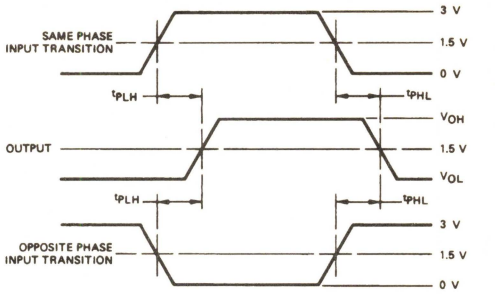
## Switching Test Waveforms




WFR02971

### Set-up, Hold, and Release Times

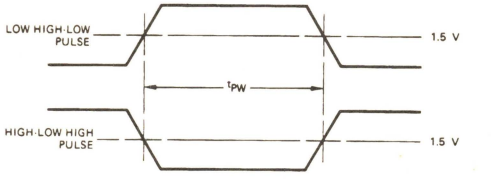
Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross hatched area is don't care condition.




WFR02980

### Propagation Delay

### Pulse Width




WFR02791

### Enable and Disable Times

| Test | $V_X$ | Output Waveform – Measurement Level |
|---|---|---|
| All $t_{PDS}$ | 5.0V | $V_{OH}$, $V_{OL}$, 1.5V |
| $t_{PHZ}$ | 0.0V | $V_{OH}$, 0.5V, 0.0V |
| $t_{PLZ}$ | 5.0V | 3V, $V_{OL}$, 0.5V |
| $t_{PZH}$ | 0.0V | $V_{OH}$, 0.0V, 1.5V |
| $t_{PZL}$ | 5.0V | 3V, 1.5V, $V_{OL}$ |

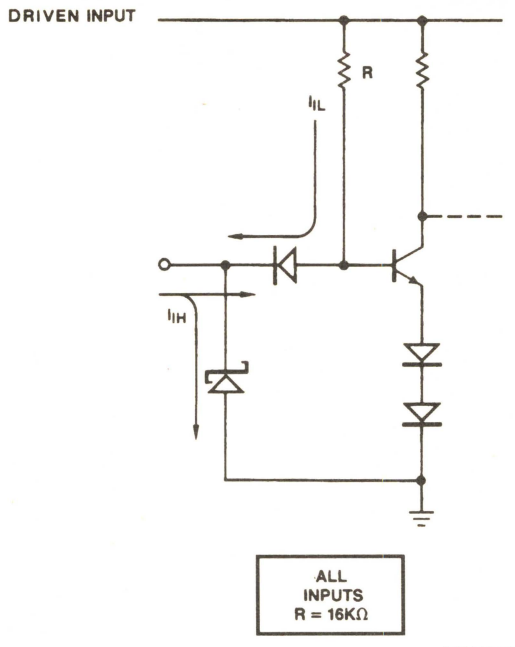
WFR02680

Notes: 1. Diagram shown for input Control Enable-LOW and input Control Disable-HIGH.
2. $S_1$, $S_2$, and $S_3$ of Load Circuit are closed except where shown.

NOTE: Pulse generator for all pulses: Rate $\leq$ 1.0 MHz; $Z_0 = 50\ \Omega$: $t_r \leq 2.5$ ns.

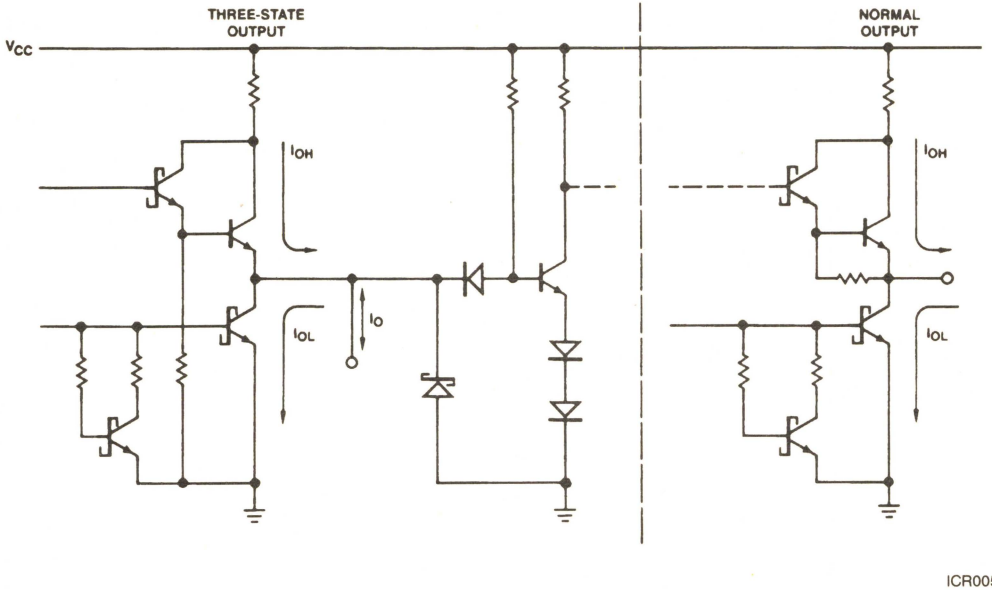
5

## Input/Output Current Interface Conditions




ICR00533

$C_O \cong 5.0$ pF, all inputs




ICR00524

$C_O \cong 5.0$ pF, all outputs

NOTE: Actual current flow direction shown.

# Am2971

Programmable Event Generator (PEG)

PRELIMINARY

## Distinctive Characteristics

- Twelve programmable, registered output waveforms
- Programmable event intervals down to 10 ns
- Multiplying phase-locked-loop (PLL) oscillator for improved timing accuracy
- Programmable clock output for system reference at 1/5 or 1/10 of internal phase-locked loop frequency
- Programmable control of timing sequence start and stop functions
- Can be clocked from either an external source or an on-board crystal oscillator

## General Description

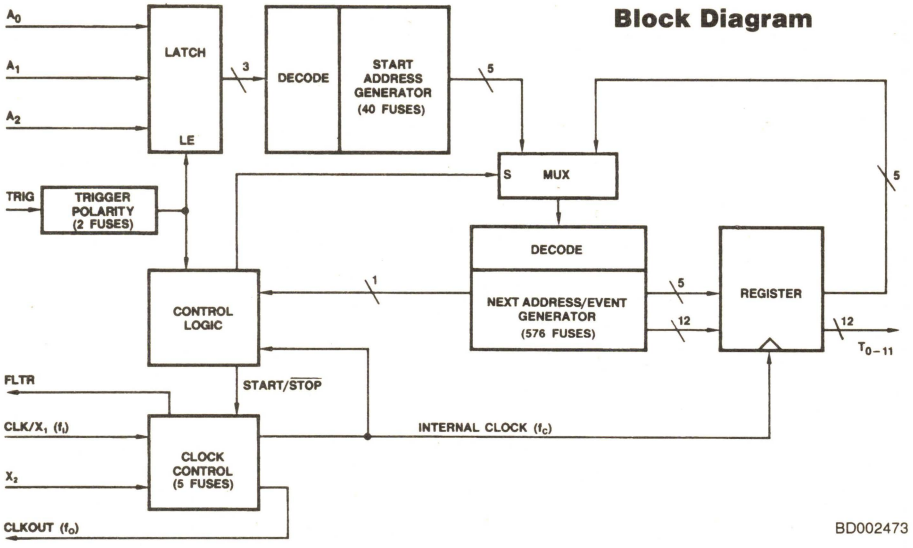
The Am2971 is a very versatile timing device. It can be used as a digital substitute for analog delay lines or as a general purpose user-programmable timing/waveform generator.

The Programmable Event Generator (PEG) has twelve independent, programmable, timing output waveforms. The resolution of the output waveforms is programmable down to 10 ns and the sequence start address is real-time user-definable. One out of eight start addresses may be selected to begin the timing sequence. The PEG can be clocked either by its 10 – 100 MHz on-chip PLL crystal oscillator (with a clock output of 1/5 or 1/10 the PLL frequency) or by the existing system clock. The chip can be clocked up to 100 MHz if the PLL is bypassed.

The frequency of the internal clock ($f_C$) can be programmed to equal 1x, 5x, 10x, 5/2x, 5/4x, 10/2x or 10/4x the PEG's input clock frequency ($f_I$). Programming the PEG is done in the same way as a PROM.

The Am2971 complements the Am2960 Dynamic Memory Support Family by allowing the user to optimize his system design for highest performance (see Figure 3).

## Block Diagram




## Related Products

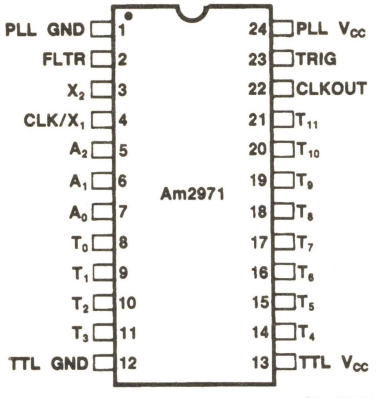
| Part No. | Description |
|---|---|
| Am2960A | Error Detection and Correction Unit |
| Am2968A | Dynamic Memory Controller |
| Am2969 | Memory Timing Controller with EDC Control |
| Am2970 | Memory Timing Controller |



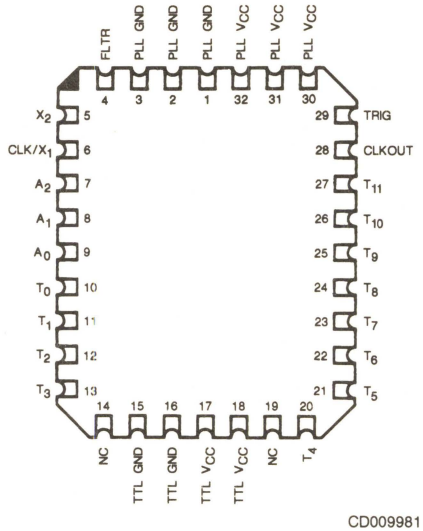
05280D/0
JANUARY 1988

## Connection Diagram

Top View



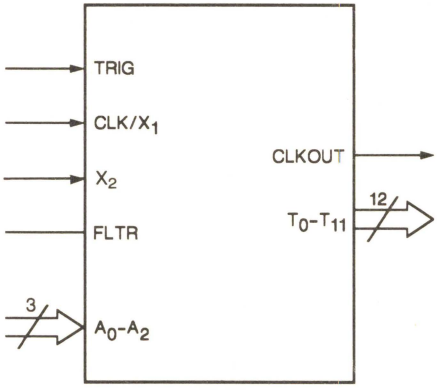

CD005852




CD009981

Note: Pin 1 is marked for orientation.

## Logic Symbol

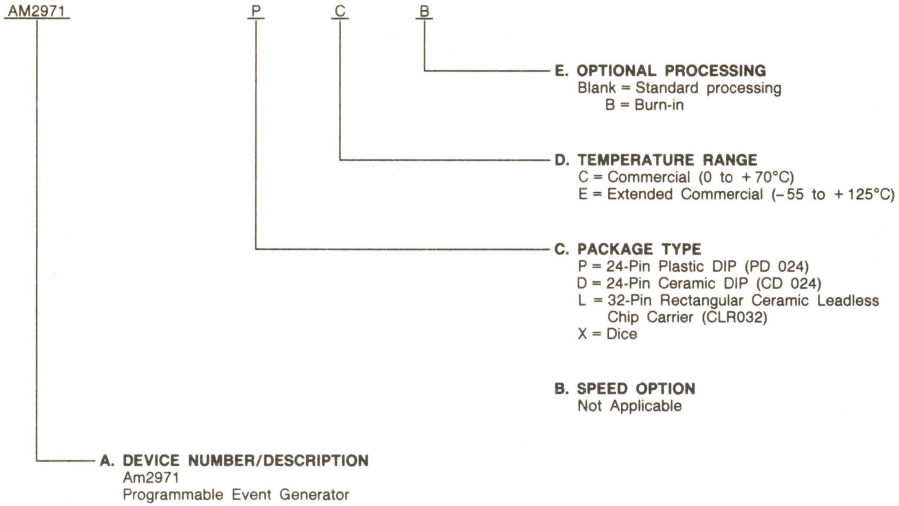



LS002711

## Ordering Information

### Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Package Type**
**D. Temperature Range**
**E. Optional Processing**




| Valid Combinations | |
|---|---|
| AM2971 | PC, PCB, DC, DCB, DE, DEB, LC, XC |

#### Valid Combinations

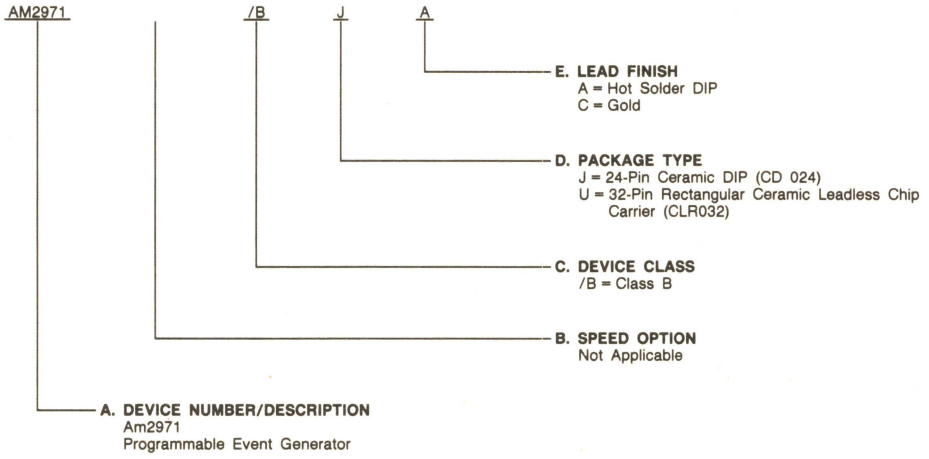
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

5

## Ordering Information

### APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. CPL (Controlled Products List) products are processed in accordance with MIL-STD-883C, but are inherently non-compliant because of package, solderability, or surface treatment exceptions to those specifications. The order number (Valid Combination) for APL products is formed by a combination of: **A. Device Number**
**B. Speed Option** (if applicable)
**C. Device Class**
**D. Package Type**
**E. Lead Finish**




| Valid Combinations | |
|---|---|
| AM2971 | /BJA, /BUC |

**Valid Combinations**

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

## Pin Description

**TRIG Trigger (Input)**
The timing cycle of the PEG can be started on either the positive or negative edge of the start (TRIG) pulse. The polarity is defined by the user as a fuse option (fuse 621) in the TRIGGER POLARITY block. The trailing edge of the start (TRIG) pulse may be used to stop the timing sequence. The STOP TRIG fuse (fuse 622) must not be blown if this option is desired. If the fuse is blown, the trailing edge of the start (TRIG) pulse will be disabled as a means of stopping the timing sequence. Instead, the PEG will search for a blown Stop Bit fuse in the Next Address/Event Generator. In the Program Mode, a high voltage level ($V_{OP}$) is applied to the TRIG input (see Figure 5).

**$A_0 - A_2$ Addresses (Inputs (3))**
These three bits define eight locations in the Start Address Generator. The Start Address Generator contains the user-programmed start locations. One of eight addresses can be selected for each cycle initiation. In the Program Mode, these inputs are unused and may be allowed to float.

**CLK/$X_1$ and $X_2$ Clock/Crystal (Input/Output)**
These two pins serve as crystal inputs $f_I$ (see Figure 1). It is recommended that an AT Cut Parallel Resonant Crystal be used. An external clock may also be applied to the CLK/$X_1$ input with the $X_2$ output left floating. Fuses 612 through 619 in the CLOCK CONTROL logic block enable the user to set the internal clock frequency as a function of the crystal or input clock frequency.

**FLTR Filter**
This pin is used to connect a 0.47-$\mu$F filter capacitor between the phase-locked-loop and ground.

**CLKOUT Output Clock (Output)**
CLKOUT, $f_O$, is a clock output pin which may be used for system reference. The output frequency for CLKOUT is either 1/5 or 1/10 the PLL clock frequency. It is fuse-programmable with the fuse located in the CLOCK CONTROL logic block. In the Program Mode, a high voltage level ($V_{IHH}$) is applied to CLKOUT for a period of time ($t_{PF}$). Applying a steady-state $V_{IHH}$ voltage to this pin for a period in excess of 400 $\mu$s is not recommended.

**$T_0 - T_{11}$ Timing Outputs (Outputs, Active HIGH)**
These are twelve timing outputs from the Next Address/Event Generator. These outputs follow a user-programmed timing pattern. Next Address/Event Generator outputs are registered allowing for glitch-free operation.

**Power, Ground TTL/PLL Power Pair**
Two power and two ground pins are required by the PEG chip. One power pair is used by the PLL (phase-locked-loop) and the internal ECL circuitry. The other power pair is used by the remainder of the chip (TTL).

## Functional Description

The Programmable Event Generator (PEG) block diagram may be divided into four blocks: Clock Control, Start Address Generator, Next Address/Event Generator and Control Logic.

Internal to the CLOCK CONTROL logic are five user-programmable fuses. One of these (fuse number 620) is used to generate the desired output frequency ($f_O$) on the CLKOUT option. The remaining four fuses (numbers 616 through 619) are used to generate the desired internal reference clock frequency ($f_C$). As shown in Figure 2, there are a variety of internal reference frequency and external CLKOUT frequency options available as a function of input frequency ($f_I$). The input frequency may be supplied either by an external source or by the internal PLL oscillator (with a crystal connected as shown in Figure 3). The reader is directed to Tables 6 and 7 for an explanation of the possible internal frequency and output frequency options in regards to user programming.

A timing sequence is initiated by a transition at the TRIG input. Two user-programmable fuses are located in the TRIGGER POLARITY block. One of the two fuses (fuse number 621) is used to define the polarity of the TRIG input. When the fuse is left unprogrammed, a timing sequence is initiated during a negative transition of the TRIG input. The second fuse in this block (fuse number 622) is used to define the end of a timing sequence. If this fuse is left unprogrammed, the timing sequence is stopped on the trailing edge of the TRIG pulse. If the fuse is programmed, the end of the timing sequence is defined by the Stop Bits as programmed into the Next Address/Event Generator functional block.

A proper transition at the TRIG input initiates a timing sequence by latching the START ADDRESS GENERATOR inputs, $A_0 - A_2$. These latched inputs are used to select one of eight different start addresses. The user defines these 5-bit start addresses by programming fuses 576 – 615 as required. Each 5-bit start address defines a starting point for the timing sequence from thirty-two possible selections in the user-defined Next Address/Event Generator block.

After the start address, subsequent addresses are generated from the information programmed into the NEXT ADDRESS/EVENT GENERATOR. As defined in Tables 3 and 8, each of the thirty-two user-defined 18-bit data strings contain three pieces of information. The first 5 bits (fuses) are used to define the next address of the desired timing sequence (out of 32 possible addresses within the Next Address/Event Generator). The next 12 bits (fuses) are used to generate output waveforms to the twelve Timing Outputs ($T_0 - T_{11}$). These 12 bits define the current logic level at each of the twelve timing ouputs. As the timing sequence progresses from address to address, these Timing Outputs will produce 12 independent waveforms, as defined by the user. Since both the address sequences and waveform logic levels are user-programmable, a wide variety of output patterns may be generated. One last bit (fuse), the Stop Bit, is used to define the end of a timing sequence.

When the Trigger Polarity Stop fuse (fuse number 622) has been programmed, the timing sequence will no longer stop on the trailing edge of the TRIG pulse. The end of a timing sequence is instead defined by the Stop Bit in each of the thirty-two Next Address/Event Generator data strings. Each Stop Bit is activated by programming. When a Timing Sequence addresses one of the thirty-two 18-bit data strings with a "1" programmed into the Stop Bit, the sequence is halted. Whenever a sequence is halted, the Timing Outputs ($T_0 - T_{11}$) will remain at the last value, as defined by the data string containing the active Stop Bit.

Each of the twelve timing waveform outputs ($T_0 - T_{11}$) from the Next Address/Event Generator has a minimum usable cycle of 40 ns (or 20 ns per change). When the PEG is operating at its maximum internal clock frequency (100 MHz), the outputs must be programmed to remain unchanged for at least two clock periods for each change of logic level. That is, although an output can only change every 20 ns minimum, the timing resolution between events may be as low as 10 ns. When the PEG's internal reference clock frequency ($f_C$) is programmed to operate at 40 MHz or slower, the timing waveform can be programmed to change on every clock.

## Oscillator

The Am2971 contains an inverting, linear amplifier which is intended to form the basis of a crystal oscillator. In designing this oscillator it is necessary to consider several factors related to the application.

The first consideration is the desired frequency accuracy. This may be subdivided into several areas. An oscillator is considered stable if it is insensitive to variations in temperature and supply voltage, and if it is unaffected by individual component changes and aging. The design of the Am2971 is such that the degree to which these goals are met is determined primarily by the choice of external components. Various types of crystals are available and the manufacturers' literature should be consulted to determine the appropriate type. For good temperature stability, zero temperature coefficient capacitors should be used (Type NPO). For extreme temperature stability, an oven must be used or some other form of temperature compensation applied.

Absolute frequency accuracy must also be considered. The resonant frequency varies with load capacitance. It is therefore important to match the load specified by the crystal manufacturer for a standard crystal (usually 32 pF), or to specify the load when ordering a special crystal. It should then be possible to determine from the crystal characteristics the load tolerance to maintain a given accuracy. If the "set-on" error due to load tolerance is unacceptable, a trimmer capacitor should be incorporated for fine adjustment.

The mechanism by which a crystal resonates is electromechanical. This resonance occurs at a fundamental frequency (1st harmonic) and at all odd harmonics of this frequency (even harmonic resonance is not mechanically possible). Unless otherwise constrained crystal oscillators operate at their fundamental frequency. However, crystals are not generally available with fundamental frequencies above 20-25 MHz. At higher frequencies, an overtone oscillator must be used. In this case, the crystal is designed to oscillate efficiently at one of its odd harmonic frequencies and additional components are included in the oscillator circuit to prevent it oscillating at lower harmonics.

Where a high degree of accuracy or stability is not required, the amplifier may be configured as an L-C oscillator. It may also be driven from an external clock source if operation is required in synchronous with that source.

## 1st Harmonic (Fundamental) Oscillator

The circuit of a typical 1st harmonic oscillator is shown in Figure 3. The crystal load is comprised of the two 68 pF capacitors in series. This 34 pF approximates the standard 32 pF crystal load. If a closer match is required then one of the capacitors should be replaced with a parallel combination of a fixed capacitor and a trimmer. The nominal value of the combination should be 60 pF to provide proper crystal loading.

A typical crystal specification for use in this circuit is:

Frequency Range: 2 – 20 MHz
Resonance: At Parallel Mode
Load: 32 pF
Stability: 0.1% or to match systems requirements
Case: H-17 — for smaller size
Temp Range: −30 to +70°C
Note: Frequency will change over temperature.

It is a good practice to ground the case of the crystal to eliminate stray pick-up and keep all connections as short as possible.

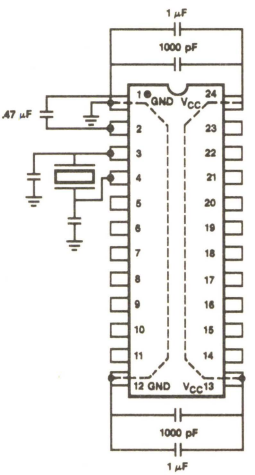
Note: At fundamental frequencies below 6 MHz it is possible for the oscillator to operate at the 3rd harmonic. To prevent this a resistor should be added in series with the $X_2$ pin as shown in the circuit diagram.

The resistor value should match the impedance of C:

$$R = X_C = \frac{1}{2\pi f C}$$

**Design Considerations (reference Figure 1)**

1. Oscillator external connections should be less than 1″ long — wirewrap is not recommended.
2. $V_{CC}$ and GND connections should be less than 1/2″ long to power plane.
3. Supply decoupling includes both high frequency and bulk storage elements.
4. The same considerations apply for 3rd overtone configurations.




PF001071

**Figure 1. Typical External Connections**

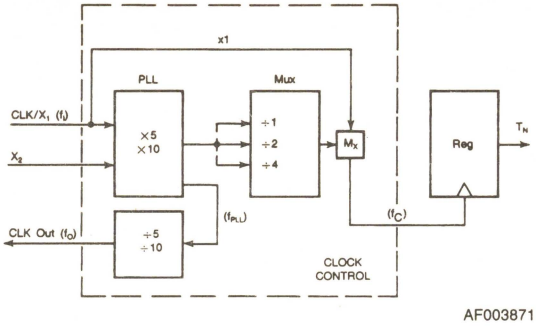
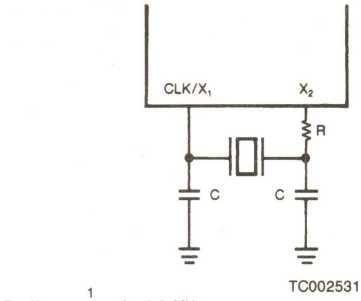



Figure 2. Clock Options




$$R = X_C = \frac{1}{2\pi f_1 C} \text{ for 2-6 MHz}$$

R = 0 for 6-20 MHz

Figure 3. Fundamental Oscillator

## Applications

The Am2971 Programmable Event Generator (PEG) is a universal, programmable digital delay line and timing/waveform generator which provides the designer an alternative to the expensive and difficult to use analog delay line. Because of the device's programmability and large number of outputs, it can replace several different delay lines at one time. The user is no longer restricted to the fixed, single event delay line. As one suggested application, this device can easily be used to generate timing for the Am2968 Dynamic Memory Controller and the Am2969 Memory Timing Controller.

AMD's Dynamic RAM Memory Support system solution is shown in Figure 4. Typically, the Timing Controller generates the Row Address Strobe Input (RASI) signal whenever a refresh or a memory cycle is requested. This signal provides a leading edge to the TRIG input of the PEG chip, thus initiating the user-defined timing sequence. Custom-tailored waveforms are generated at the Timing Outputs ($T_0 - T_{11}$), providing proper and precise sequencing between the Am2968 and Am2969 for optimum system performance.

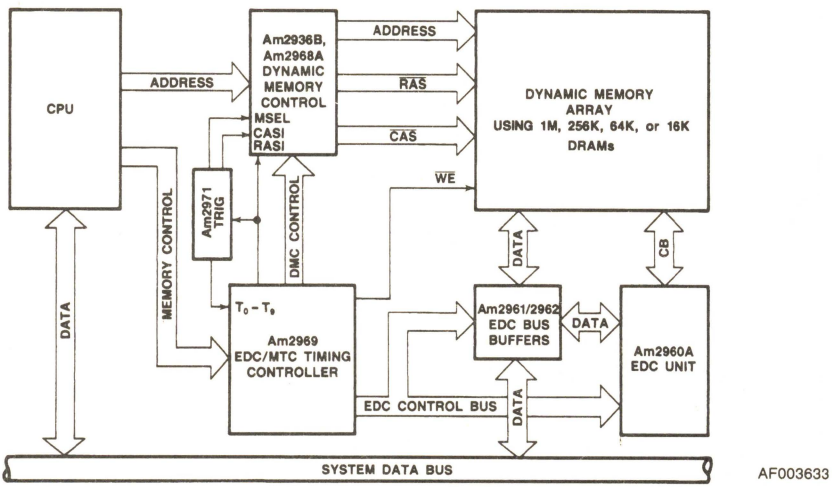



Figure 4. 16-Bit High-Performance Computer Memory System Application

## Programmer/Development Systems

Refer to Programmer Reference Guide.

## Absolute Maximum Ratings

Storage Temperature ..... −65°C to +150°C
Ambient Temperature with
Power Applied ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continuous (TTL $V_{CC}$ and PLL $V_{CC}$) ..... 0 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... 0 V to +$V_{CC}$ max.
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −18 mA to +5.0 mA

*Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.*

## Operating Ranges

Commercial (C) Devices
Temperature ($T_A$) ..... 0 to +70°C
TTL $V_{CC}$ and PLL $V_{CC}$ ..... 5.0 V ±10%
Min. ..... 4.50 V
Max. ..... 5.50 V

Extended Commercial (E) or Military* (M) Devices
Temperature ($T_C$) ..... −55 to +125°C
TTL $V_{CC}$ and PLL $V_{CC}$ ..... 5.0 V ±10%
Min. ..... 4.50 V
Max. ..... 5.50 V

*Operating ranges define those limits between which the functionality of the device is guaranteed.*

*Military Product 100% tested at $T_C$ = +25°C, +125°C, and −55°C.

## DC Characteristics over operating range unless otherwise specified

| Parameter Symbol | Parameter Description | Test Conditions | | | Min. | Max. | Units |
|---|---|---|---|---|---|---|---|
| $V_{OH}$ | Output HIGH Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OH}$ = −1 mA | COM'L. | 2.7 | | V |
| | | | | MIL. | 2.5 | | |
| $V_{OL}$ | Output LOW Voltage | $V_{CC}$ = Min., $V_{IN}$ = $V_{IH}$ or $V_{IL}$ | $I_{OL}$ = 8 mA | | | 0.4 | V |
| $V_{IH}$ (Note 1) | Input HIGH Voltage | Guaranteed Input HIGH Voltage for All Inputs | | | 2.0 | | V |
| $V_{IL}$ (Note 1) | Input LOW Voltage | Guaranteed Input LOW Voltage for All Inputs | | | | 0.8 | V |
| $V_{IHC}$ | Input HIGH Voltage to CLK/$X_1$ | Guaranteed Input HIGH Voltage for All Inputs | | | 3.0 | | V |
| $V_{ILC}$ | Input LOW Voltage to CLK/$X_1$ | Guaranteed Input LOW Voltage for All Inputs | | | | 0.8 | V |
| $V_I$ (Note 1) | Input Clamp | $V_{CC}$ = Min. | $I_{IN}$ = −18 mA | | | −1.2 | V |
| $I_{IH}$ | Input HIGH Current | $V_{CC}$ = Min., $V_{IN}$ = 3.0 V | CLK/$X_1$, $X_2$ | | | 700 | µA |
| | | $V_{CC}$ = Max., $V_{IN}$ = 2.7 V | $A_0$ − $A_2$ and TRIG | | | 20 | |
| $I_{IL}$ | Input LOW Current | $V_{CC}$ = Max., $V_{IN}$ = 0.5 V | CLK/$X_1$, $X_2$ | | | −500 | µA |
| | | $V_{CC}$ = Max., $V_{IN}$ = 0.5 V | $A_0$ − $A_2$ and TRIG | | | −250 | |
| $I_I$ | Input Current | $V_{CC}$ = Min., $V_{IN}$ = 4.0 V | CLK/$X_1$, $X_2$ | | | 1 | mA |
| | | $V_{CC}$ = Max., $V_{IN}$ = 5.5 V | $A_0$ − $A_2$ | | | 100 | µA |
| | | $V_{CC}$ = Max., $V_{IN}$ = $V_{CC}$ − 0.5 | TRIG | | | 100 | |
| $I_{SC}$ (Note 2) | Output Short Circuit Current | $V_{CC}$ = Max., $V_{IN}$ = 0 V | $V_O$ = 0 V | | −15 | −100 | mA |
| $I_{CC}$ (Note 3) | Power Supply Current | $V_{CC}$ = Max. | T = −55°C, 0°C, +25°C | | | 310 | mA |
| | | | T = +70°C, +125°C | | | 240 | |

Notes: 1. Does not apply to CLK/$X_1$ and $X_2$.
2. No more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
3. $I_{CC}$ varies with temperature and oscillation frequency.

## Switching Characteristics over operating range unless otherwise specified

| No. | Parameter Symbol | Parameter Description | Test Conditions | Min. | Max. | Units |
|---|---|---|---|---|---|---|
| 1 | FREQ IN @ CLK/$X_1$ | CLK/$X_1$ Input Clock Frequency (Note 2) | a) PLL Frequency Multiplication Mode (TTL Input @ CLK/$X_1$) | 10 | 70 | MHz |
| | | | b) PLL Frequency Multiplication Mode (Crystal @ CLK/$X_1$, $X_2$) | | | |
| | | | c) Flow-through Mode (PLL bypassed) (TTL Input @ CLK/$X_1$) | 0 | 85 | |
| | | | d) Flow-through Mode (PLL bypassed) (Crystal @ CLK/$X_1$, $X_2$) | | | |
| 2 | $t_{RISE}$ @ CLKOUT | CLKOUT Rise Time (Note 2) | | | 14 | ns |
| 3 | $t_{FALL}$ @ CLKOUT | CLKOUT Fall Time (Note 2) | | | 10 | ns |
| 4 | $t_{RISE}$ @ $T_0 - T_{11}$ | $T_0 - T_{11}$ Rise Time (Note 2) | | | 10 | ns |
| 5 | $t_{FALL}$ @ $T_0 - T_{11}$ | $T_0 - T_{11}$ Fall Time (Note 2) | | | 9 | ns |
| 6 | $t_{SKEW}$ @ $T_0 - T_{11}$ | Skews between the $T_0 - T_{11}$ Outputs (LOW-to-HIGH Transition) | $C_L$ = 50 pF | | 3.5 | ns |
| 7 | $t_{SKEW}$ @ $T_0 - T_{11}$ | Skews between the $T_0 - T_{11}$ Outputs (HIGH-to-LOW Transition) | $C_L$ = 50 pF | | 5.0 | ns |
| 8 | $t_{SKEW}$ @ $T_0 - T_{11}$ | Skews between the $T_0 - T_{11}$ Outputs (Mixed Transition) | $C_L$ = 50 pF | | 8.5 | ns |
| 9 | $t_{SET}$ TRIG to CLK/$X_1$ | TRIG ⌐ to CLK/$X_1$ ⌐ Setup Time | a) PLL Frequency Multiplication Mode (Note 2) | | 11 | ns |
| | | | b) Flow-through Mode (PLL bypassed) | | 11 | ns |
| 10 | $t_{SET}$ TRIG to CLK/$X_1$ | TRIG ¬ to CLK/$X_1$ ⌐ Setup Time | a) PLL Frequency Multiplication Mode (Note 2) | | 6 | ns |
| | | | b) Flow-through Mode (PLL bypassed) | | 4 | ns |
| 11 | $t_{PD}$ CLK/$X_1$ to $T_0 - T_{11}$ | Propagation Delay from CLK/$X_1$ ⌐ to the $T_0 - T_{11}$ Outputs | a) PLL Frequency Multiplication Mode (Note 2) | | 25 | ns |
| | | | b) Flow-through Mode (PLL bypassed) | | 23 | ns |
| 12 | $t_{PD}$ CLK/$X_1$ to CLKOUT | Propagation Delay from CLK/$X_1$ to the CLKOUT Outputs | PLL Frequency Multiplication Mode (Note 2) | | 17.5 | ns |
| 13 | $t_{SET}$ $A_0 - A_2$ to TRIG | $A_0 - A_2$ Inputs to TRIG ⌐ (Note 3) | PLL Frequency Multiplication Mode/Flow-through Mode (PLL bypassed) | | 1.0 | ns |
| 14 | $t_{HOLD}$ $A_0 - A_2$ to TRIG | $A_0 - A_2$ Inputs to TRIG ⌐ Hold Time (Note 3) | PLL Frequency Multiplication Mode/Flow-through Mode (PLL bypassed) | | 11.0 | ns |

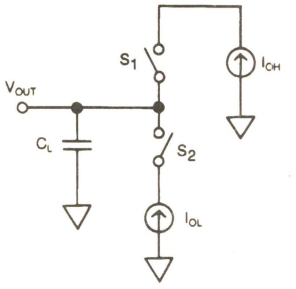
### Calculated Switching Characteristics

| | | | | |
|---|---|---|---|---|
| 15 | $t_{PERIOD}$ Resolution @ $T_0 - T_{11}$ | Timing Resolution between the $T_0 - T_{11}$ Outputs | | $1/f_I$ (1a or 1c) |
| 16 | $t_{PWH}$ @ TRIG (Calculated) | TRIG Input Pulse Width (HIGH State) (Note 1) | a) PLL Frequency Multiplication Mode | $t_{SET}$ (9a) + $1/f_C$ + 5 |
| | | | b) Flow-through Mode (PLL bypassed) | $t_{SET}$ (9b) + $1/f_I$ + 5 |
| 17 | $t_{PWL}$ @ TRIG (Calculated) | TRIG Input Pulse Width (LOW State) (Note 1) | a) PLL Frequency Multiplication Mode | $1/f_C$ |
| | | | b) Flow-through Mode (PLL bypassed) | $t_{SET}$ (10b) |
| 18 | $t_{PD}$ TRIG $T_0 - T_{11}$ (Calculated) | Delay from TRIG ⌐ to $T_0 - T_{11}$ Active | a) PLL Frequency Multiplication Mode | $t_{SET}$ (9a) + $1/f_C$ + $t_{PD}$ (11a) |
| | | | b) Flow-through Mode (PLL bypassed) | $t_{SET}$ (9b) + $1/f_I$ + $t_{PD}$ (11b) |
| 19 | $t_{PD}$ TRIG $T_0 - T_{11}$ (Calculated) | Delay from TRIG ¬ to $T_0 - T_{11}$ Inactive | a) PLL Frequency Multiplication Mode | $t_{SET}$ (10a) + $1/f_C$ + $t_{PD}$ (11a) |
| | | | b) Flow-through Mode (PLL bypassed) | $t_{SET}$ (10b) + $1/f_I$ + $t_{PD}$ (11b) |

Notes: 1. Not Tested; calculated using other parameters.
2. Not Tested; correlated.
3. Only $A_2$ is tested.

5

## Switching Test Circuit

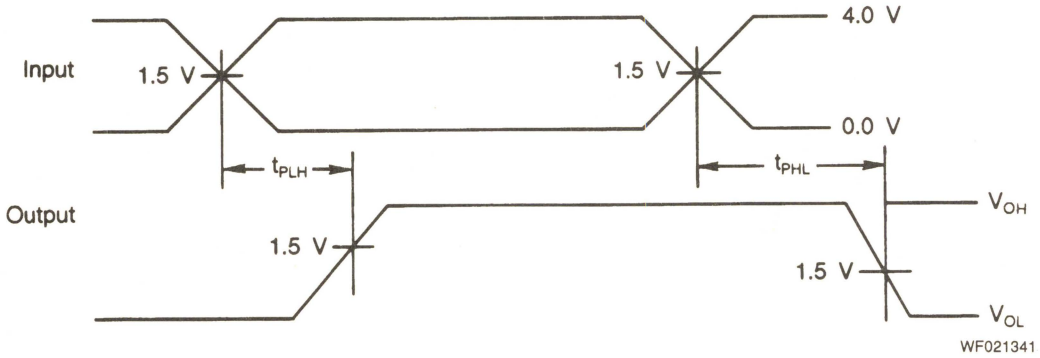



TC003132

**A. Outputs**

Notes: 1. $C_L$ = 50 pF, the load capacitance includes scope probe, wiring, and stray capacitance without the device in the test fixture.
2. $S_1$ and $S_2$ are open during all DC and functional testing
3. During AC testing, switches are set as follows:
   1) For $V_{OUT} > 1.5$ V, $S_1$ is closed and $S_2$ open
   2) For $V_{OUT} < 1.5$ V, $S_1$ is open and $S_2$ closed

## Switching Test Waveform



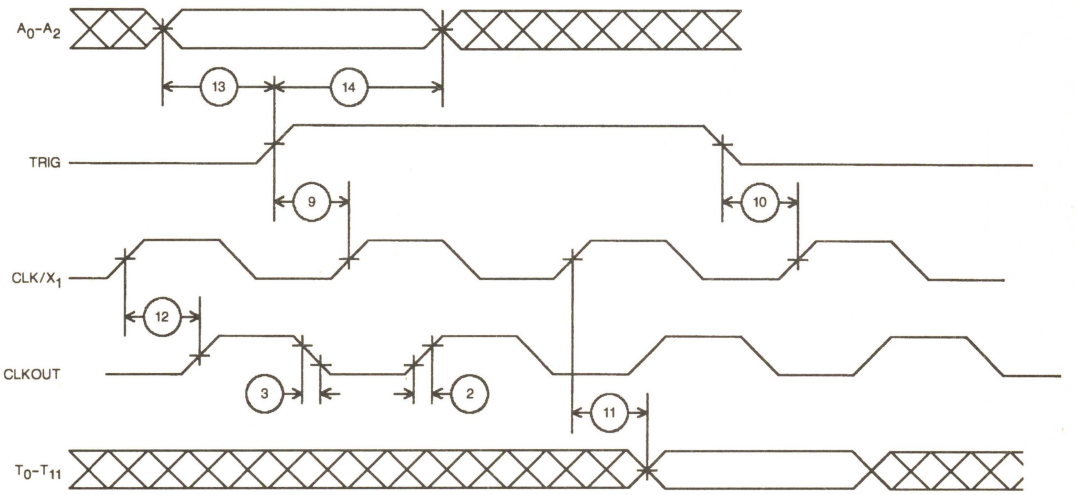

WF021341

## Key to Switching Waveforms

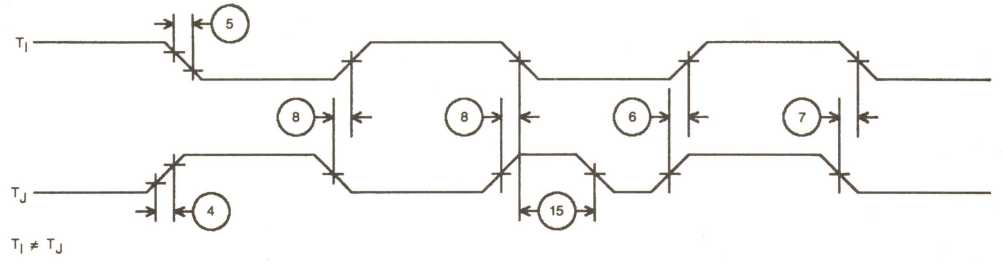
| WAVEFORM | INPUTS | OUTPUTS |
|---|---|---|
| ——— | MUST BE STEADY | WILL BE STEADY |
| ⧵⧵⧵ | MAY CHANGE FROM H TO L | WILL BE CHANGING FROM H TO L |
| ⧸⧸⧸ | MAY CHANGE FROM L TO H | WILL BE CHANGING FROM L TO H |
| ⧸⧵⧸⧵ | DON'T CARE; ANY CHANGE PERMITTED | CHANGING; STATE UNKNOWN |
| ⋈ | DOES NOT APPLY | CENTER LINE IS HIGH IMPEDANCE "OFF" STATE |

KS000010

## Switching Waveforms



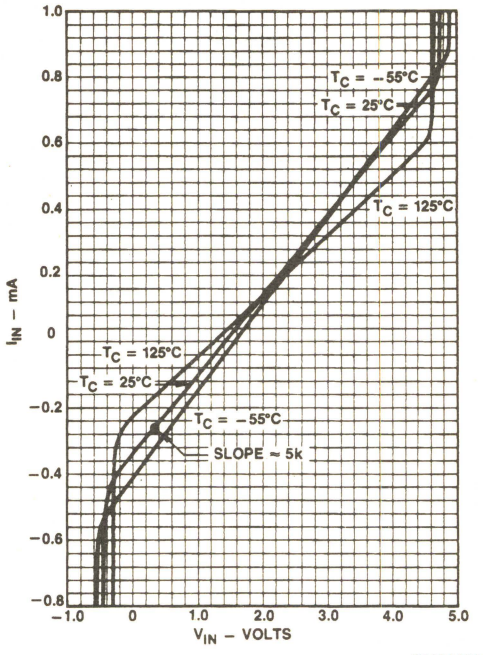

WF022521




WF022530

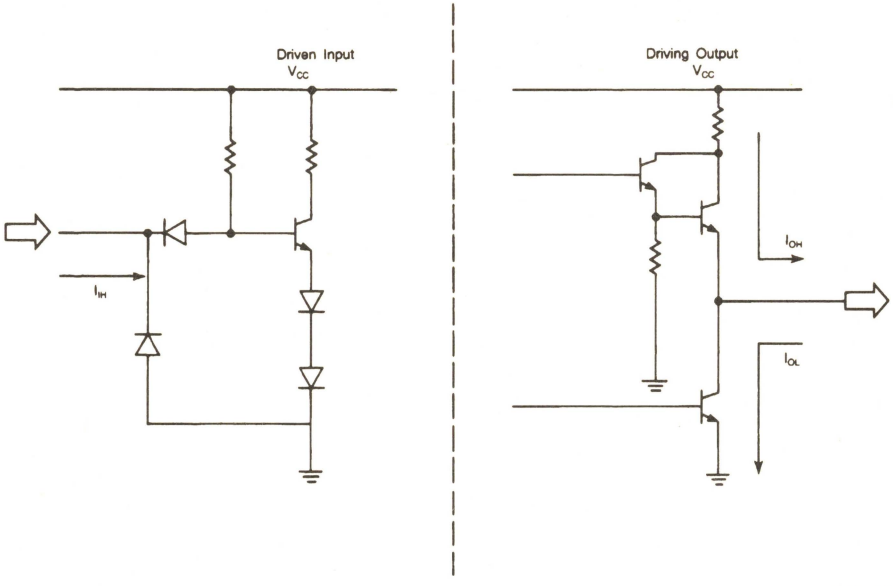
**Rise Time/Fall Time/Skews**

## Typical Performance Curve

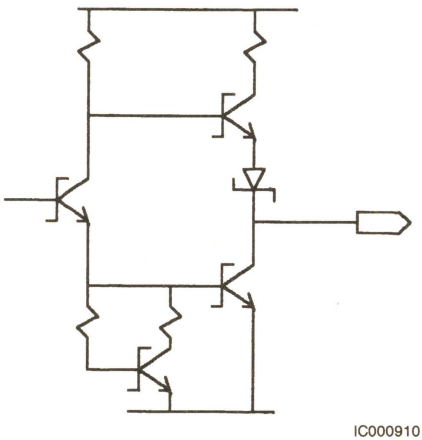



**CLK/$X_1$ Crystal Input Characteristics**

## Input/Output Circuit Diagrams




IC000881

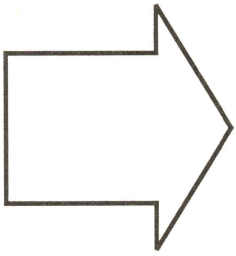


IC000910

**Output Configuration for CLKOUT**

5


IMPORTANT
NOTES



## Data Sheets

- TTL/CMOS PAL Devices
- TTL/CMOS AmPAL Devices
- PROSE/PLS Sequencers
- FPC/PEG Sequencers
- ECL PAL Devices
- HAL/ZHAL Devices
- Military PAL Devices
- Logic Cell Array
- Electrical Definitions


IMPORTANT
NOTES

# PAL10H20EV/EG8
# PAL10020EV/EG8

## ADVANCE INFORMATION

### Features/Benefits

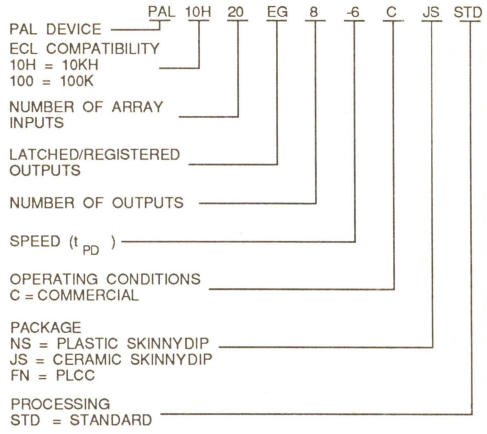
- 20 logic inputs: 12 external, 8 feedback
- 8 outputs programmable as registered, latched, or combinatorial
- ECL technology provides 6 ns propagation delay
- 64 product terms
- 10KH and 100K compatible versions
- Space saving 24-pin SKINNYDIP® package
- Programmable using standard TTL programmers with adapter
- Security fuse prevents unauthorized copying

### Description

The PAL10H/10020EV/EG8 is a high density universal ECL PAL device. Outputs can be specified as registered or combinatorial for the 20EV8, and latched or combinatorial for the 20EG8, on an individual basis. In addition, varied product term distribution allows up to twelve product terms per output. These features allow complex designs at the high speed of ECL technology.

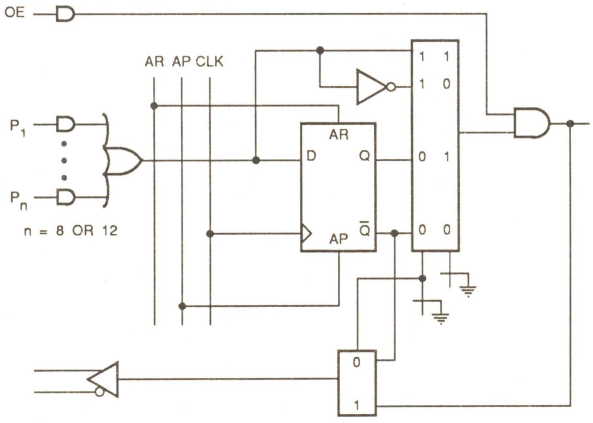
Other features include asynchronous preset and reset, and individual output enables. Programmable polarity allows either active-HIGH or active-LOW outputs.

### Ordering Information




630 01

### Macrocell




630 02

**PAL10H/10020EV8 Macrocell**

# Programmable Array Logic ECL PAL® Device

# PAL10H20G8

## Features/Benefits

- 20 logic inputs: 12 external, 8 feedback
- 8 latched outputs
- Programmable latch bypass
- ECL technology provides 4.5 ns $t_{SU}$, 6 ns $t_{PD}$
- 32 product terms with term sharing
- 10KH ECL compatible
- 50-Ω termination drive
- Input pull-down resistors
- Voltage compensated
- Space-saving 24-pin SKINNYDIP® and 28-pin PLCC packages
- Programmable using standard TTL programmers with adapter
- Greater than 99% programming yield
- Security fuse prevents unauthorized copying

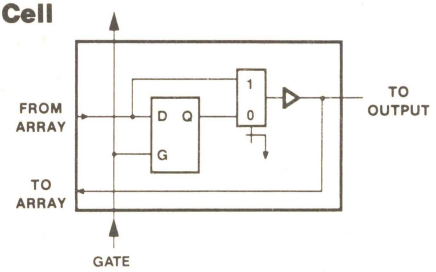
## Description

The PAL10H20G8 is a 10KH family compatible ECL PAL device having twelve dedicated inputs and eight latched outputs with feedback. A programmable AND array and a fixed OR array make possible the implementation of a wide variety of logic functions with far fewer packages than with SSI devices. The logic is implemented by opening metal fuse connections within the AND array. Designs can be specified by using any of a variety of software packages which accept the design and assemble a file that can be downloaded into a device programmer. The device can be programmed using any of the qualified PAL device programmers (refer to the Programmer Reference Guide).

The output latches will hold data when the gate (G) pin is high, and will be transparent when the gate is low. The outputs can drive a 50 Ω termination to $V_{CC}$ - 2.0 V.

The input pins have 50 kΩ internal pull-down resistors, which allow unused inputs to be left open. Open inputs will assume a logic low state.

## Latch Cell




## Features

Each output latch has a bypass fuse, for creating a combinatorial output. There are two gate pins, each of which drives a bank of four output latches. If all of the outputs in a bank are programmed to have their latches bypassed, the gate pin for that bank can be used as an input to the array.

The programmable AND array contains a total of thirty-two product terms. Product terms are arranged in groups of eight. The terms in each group can be shared mutually exclusively between two adjacent output cells. If a particular product term is needed for two outputs, then two identical product terms are generated: one for each output.

A security fuse is provided to help protect the fuse pattern from unauthorized copying. Once the security fuse has been programmed, it is no longer possible to verify the contents of the fuse array electrically. The security fuse has no effect on functionality.

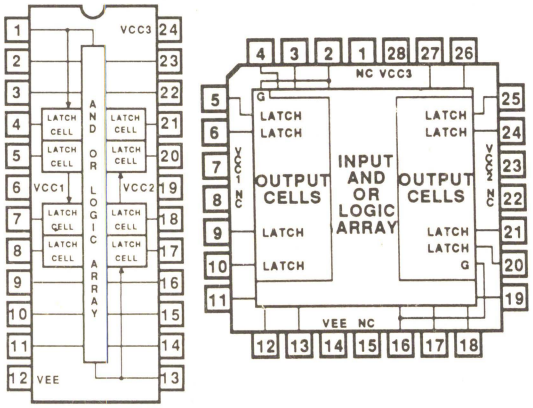
Note that if all latches within a bank have not been bypassed, an attempt to use the gate pin as an array input can cause input setup time violations, and may be flagged as an error when assembling the equations.

## Ordering Information

PAL10H20G8 NS STD

- PROGRAMMABLE ARRAY LOGIC FAMILY
- ECL COMPATIBILITY: 10H = 10KH
- NUMBER OF ARRAY INPUTS
- OUTPUT TYPE: G = Latched
- NUMBER OF OUTPUTS
- PACKAGE: NS = Plastic SKINNYDIP; JS = Ceramic SKINNYDIP; FN = PLCC
- PROCESSING: STD = Standard; XXXX = Special Processing

## Pin Configurations




10340A
JANUARY 1988

## Absolute Maximum Ratings

These ratings specify the conditions above which the device may be permanently damaged. AC and DC specifications are not necessarily guaranteed over this range.

Supply voltage $V_{EE}$ ($V_{CC1}$ = $V_{CC2}$ = $V_{CC3}$ = 0 Volts) ..... -8 V to 0 V
Input voltage $V_I$ ($V_{CC1}$ = $V_{CC2}$ = $V_{CC3}$ = 0 Volts) ..... 0 V to $V_{EE}$
Output current, $I_{OUT}$
- Continuous ..... 35 mA
- Surge ..... 100 mA

Storage temperature range, $T_{stg}$ ..... -65°C to 150°C
Maximum burn-in temperature
- Ceramic package ..... 125°C
- Plastic package ..... 75°C

## Operating Conditions

| SYMBOL | PARAMETER | COMMERCIAL MIN | NOM | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{EE}$ | Supply voltage ($V_{CC}$ = 0 V) | -5.46 | -5.2 | -4.94 | V |
| $T_A$ | Operating free-air temperature | 0 | | 75 | °C |
| $t_{su}$ | Latch input setup time | 4.5 | | | ns |
| $t_h$ | Latch input hold time | 0 | | | ns |
| $t_w$ | Gate signal pulse width | 2.0 | | | ns |

## Electrical Characteristics $V_{EE}$ = -5.2 V ±5% (See note 1)

| SYMBOL | PARAMETER | TEST CONDITIONS | 0° MIN | 0° MAX | 25° MIN | 25° MAX | 75° MIN | 75° MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $I_{EE}$ | Power supply current | Inputs $V_{IN}$ = $V_{IH}$ Max | — | 225 | — | 225 | — | 225 | mA |
| $I_{inH}$ | Input current high | $V_{IH}$ Min < $V_{in}$ < $V_{IH}$ Max | — | 425 | — | 265 | — | 265 | μA |
| $I_{inL}$ | Input current low | $V_{IL}$ Min < $V_{in}$ < $V_{IL}$ Max | 0.5 | — | 0.5 | — | 0.3 | — | μA |
| $V_{OH}$ | High output voltage | (See Note 2) | -1.02 | -0.84 | -0.98 | -0.81 | -0.92 | -0.735 | V dc |
| $V_{OL}$ | Low output voltage | (See Note 2) | -1.95 | -1.63 | -1.95 | -1.63 | -1.95 | -1.60 | V dc |
| $V_{IH}$ | High input voltage | (See Note 2) | -1.17 | -0.84 | -1.13 | -0.81 | -1.07 | -0.735 | V dc |
| $V_{IL}$ | Low input voltage | (See Note 2) | -1.95 | -1.48 | -1.95 | -1.48 | -1.95 | -1.45 | V dc |

## Switching Characteristics $V_{EE}$ = -5.2 V ±5% (See note 1)

| SYMBOL | PARAMETER | 0° MIN | 0° MAX | 25° MIN | 25° MAX | 75° MIN | 75° MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Propagation delay | 2.0 | 6.0 | 2.0 | 6.0 | 2.0 | 6.0 | ns |
| $t_R$ | Rise time (20%-80%) | 0.8 | 2.2 | 0.7 | 2.0 | 0.8 | 2.2 | ns |
| $t_F$ | Fall time (80%-20%) | 0.8 | 2.2 | 0.7 | 2.0 | 0.8 | 2.2 | ns |
| $t_G$ | Gate to output delay | 1.0 | 2.5 | 1.0 | 2.5 | 1.0 | 2.5 | ns |

Note: 1. Each ECL 10KH series circuit has been designed to meet the specifications shown in test table, after thermal equilibrium has been established. The circuit is in a test socket or mounted on a printed circuit board and transverse air flow greater than 500 linear fpm is maintained.

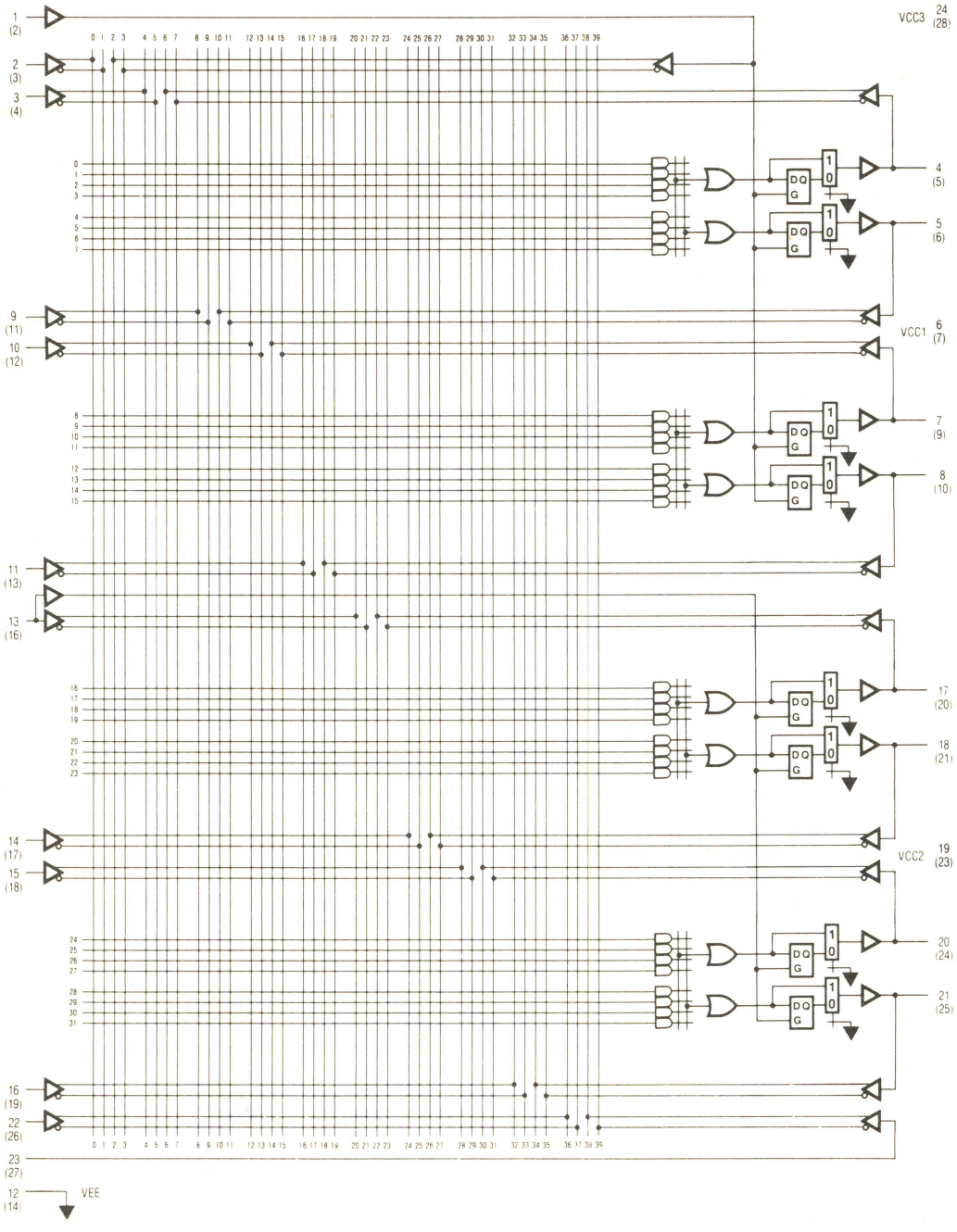
2. Outputs are terminated through a 50-Ω resistor to $V_{CC}$ -2.0 V.

3. If pin 14 (PLCC pin 17) is not used, it should be left open or terminated to $V_{TT}$ (= $V_{CC}$ - 2.0 V). It should not be terminated to $V_{EE}$.

## General Information

(refer to page 5-389)

## Logic Diagram




Note: Numbers in parentheses refer to the PLCC pin number.
PLCC pins 1, 8, 15, and 22 are not connected.

## Using the PAL10H20G8 Device as a State Machine

Latches can be used in the implementation of state machines, but care must be taken in their use. They cannot be treated as if they were registers, which are more common in the TTL PAL devices. Since a latch is a level-sensitive storage device, it is more difficult to control the sequencing of states. In theory, when the gate pin is lowered, the latch can assume a new state. After waiting for the state to stabilize and for the feedback signal to propagate, the new state can be latched. Latching the state delays any further state changes until the next time the gate signal G goes low.

## Latches Are Not Registers

The danger in treating a latch just like a register is that a feedback race condition will be explicitly built into the circuit. Enough setup time must be allowed for latching new data, yet if too much delay is allowed, the transparent latch may actually change state twice before being latched. For example, a divide-by-two counter (Figure 1a) may oscillate until the gate pin is raised if too much setup time is provided. The final state will depend on how fast the output was oscillating (Figure 1b), and will be unpredictable.

## Use a Dual-Phase Clock

The usual method of dealing with this problem is to use a dual-phase clock, with two sets of latches. The logic must be partitioned such that all latches that are enabled on one phase of the clock feed only latches that are enabled on the opposite phase. This allows operation in a master-slave mode. The implementation can be made by providing one latch merely as a holding element, which can pass the data unchanged to another latch; this is essentially a master-slave register. In such a situation, two latches are needed for each state bit. The divide-by-two counter using a master and a slave is shown in Figure 1c.

If the entire state machine can be partitioned into two groups of state bits such that group 1 states only feed group 2 states and vice versa, then it becomes possible to place logic between the master latch and the slave latch. At that point the terms "master" and "slave" lose significance; each latch may implement a state bit by itself. The minimum timing is illustrated in Figure 2.

## Greater System Speed And Efficiency

This kind of arrangement can sometimes be used to obtain a greater overall system speed than would be possible using

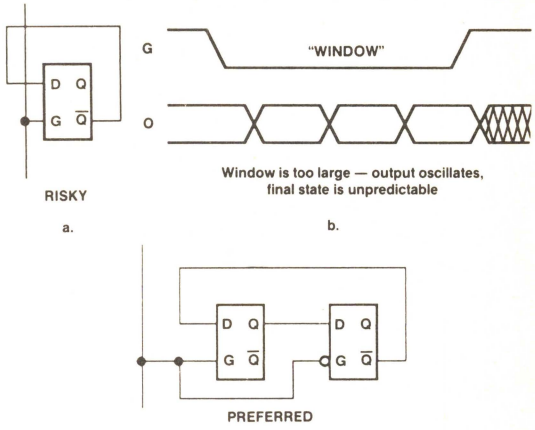



Figure 1

registers instead of latches. Since a register is essentially made up of two latches, it can also provide for a more efficient design with latches.

## The PAL10H20G8 Has Two Gate Pins

The PAL10H20G8 device is equipped with two external gate lines, each of which controls a bank of four latches. This makes it possible to split a clock signal and feed opposite phases to these clock lines. The gate pins have a $t_{su}$ delay in order to guarantee that an output will change after a time $t_G$ from when the gate pin is lowered. Referring then to Figure 2, we have:

$$f_{MAX} = \frac{1}{2t_{su} + 2t_G}$$

Note that the gate pins should not be used as array inputs if they are also being used to enable the latches, since there is a great risk (or even certainty) of violating the necessary input setup times.

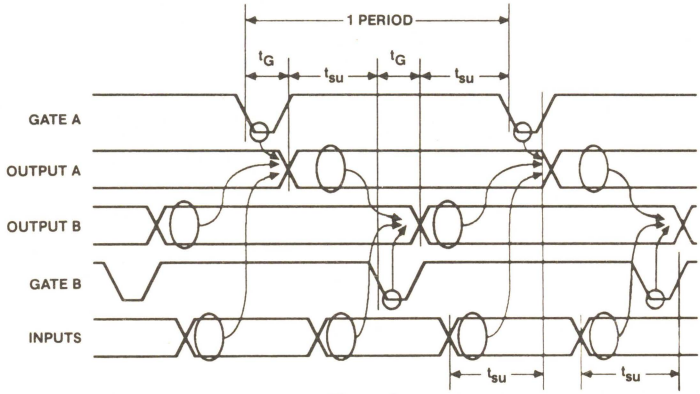



Figure 2

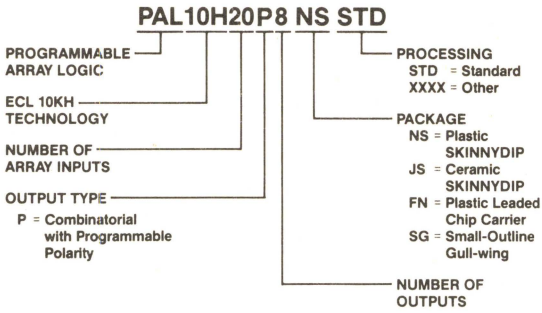
# Combinatorial ECL PAL Device

# PAL10H20P8

## Features/Benefits

- 20 logic inputs: 12 external, 8 feedback
- 8 outputs with programmable polarity
- ECL technology for ultra-high speed—max $t_{PD}$ = 6 ns
- 32 product terms with term sharing
- 10 KH ECL compatible
- Fully AC tested
- Input pull-down resistors
- Voltage compensated
- Space-saving 24-pin SKINNYDIP® and 28-pin PLCC packages
- Programmable using standard TTL programmers with adapter
- Greater than 99% programming yield
- Security fuse prevents unauthorized copying

## Ordering Information




## Description

The PAL10H20P8 is a 10KH family compatible ECL PAL device having twelve dedicated inputs and eight outputs with feedback. A programmable AND array and a fixed OR array make possible the implementation of a wide variety of logic functions with far fewer packages than with SSI devices. The logic is implemented by opening metal fuse connections within the AND array. Designs can be specified by using any of a variety of software packages which accept the design and assemble a file that can be downloaded into a device programmer. The device can be programmed using any of the qualified PAL device programmers (refer to the Programmer Reference Guide).

The outputs are equipped with programmable polarity. They can drive a 50-Ω termination (to $V_{CC}$-2.0 V). Product term sharing is provided to allow greater flexibility in assigning product terms to outputs.

The input pins have 50-kΩ internal pull-down resistors, which allow unused inputs to be left open. Open inputs will assume a logic low state.

## Features

Each output has a programmable polarity fuse, allowing for more efficient representation of many logic functions. Each output is active high with polarity fuse intact, and active low with the polarity fuse blown.

The programmable AND array contains a total of thirty-two product terms. Product terms are arranged in groups of eight. The terms in each group can be shared mutually exclusively between two adjacent output cells. If a particular product term is needed for two outputs, then two identical product terms are generated: one for each output.

A security fuse is provided to help protect the fuse pattern from unauthorized copying. Once the security fuse has been programmed, it is no longer possible to verify the contents of the fuse array electrically. The security fuse has no effect on functionality.

## Packages

The PAL10H20P8 is available in the plastic SKINNYDIP (NS), ceramic SKINNYDIP (JS), and plastic leaded chip carrier (FN) packages. For drawings, refer to PAL Device Package Outlines.

10346A
JANUARY 1988

## Absolute Maximum Ratings

These ratings specify the conditions above which the device may be permanently damaged. AC and DC specifications are not necessarily guaranteed over this range.

Supply voltage $V_{EE}$ ($V_{CC1}$ = $V_{CC2}$ = $V_{CC3}$ = 0 V) ..... -8.0 V to 0 V
Input voltage $V_I$ ($V_{CC1}$ = $V_{CC2}$ = $V_{CC3}$ = 0 V) ..... 0 V to $V_{EE}$
Output current, $I_{OUT}$:
Continuous ..... 35 mA
Surge ..... 100 mA
Storage temperature range, $T_{stg}$ ..... -65°C to 150°C

## Operating Conditions

| SYMBOL | PARAMETER | COMMERCIAL MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{EE}$ | Supply voltage ($V_{CC}$ = 0 V) | -5.46 | -5.2 | -4.94 | V |
| $T_A$ | Operating free-air temperature | 0 | | 75 | °C |

## Electrical Characteristics $V_{EE}$ = -5.2 V ±5% (See note 1)

| SYMBOL | PARAMETER | TEST CONDITIONS | 0°C MIN | 0°C MAX | 25°C MIN | 25°C MAX | 75°C MIN | 75°C MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $I_{EE}$ | Power supply current | Inputs $V_{IN}$ = $V_{IH}$ MAX | — | 210 | — | 210 | — | 210 | mA |
| $I_{inH}$ | Input current high | $V_{IH}$ MIN < $V_{in}$ < $V_{IH}$ MAX | — | 425 | — | 265 | — | 265 | μA |
| $I_{inL}$ | Input current low | $V_{IL}$ MIN < $V_{in}$ < $V_{IL}$ MAX | 0.5 | — | 0.5 | — | 0.3 | — | μA |
| $V_{OH}$ | High output voltage | (See note 2) | -1.02 | -0.84 | -0.98 | -0.81 | -0.92 | -0.735 | $V_{dc}$ |
| $V_{OL}$ | Low output voltage | (See note 2) | -1.95 | -1.63 | -1.95 | -1.63 | -1.95 | -1.60 | $V_{dc}$ |
| $V_{IH}$ | High input voltage | (See note 2) | -1.17 | -0.84 | -1.13 | -0.81 | -1.07 | -0.735 | $V_{dc}$ |
| $V_{IL}$ | Low input voltage | (See note 2) | -1.95 | -1.48 | -1.95 | -1.48 | -1.95 | -1.45 | $V_{dc}$ |

5

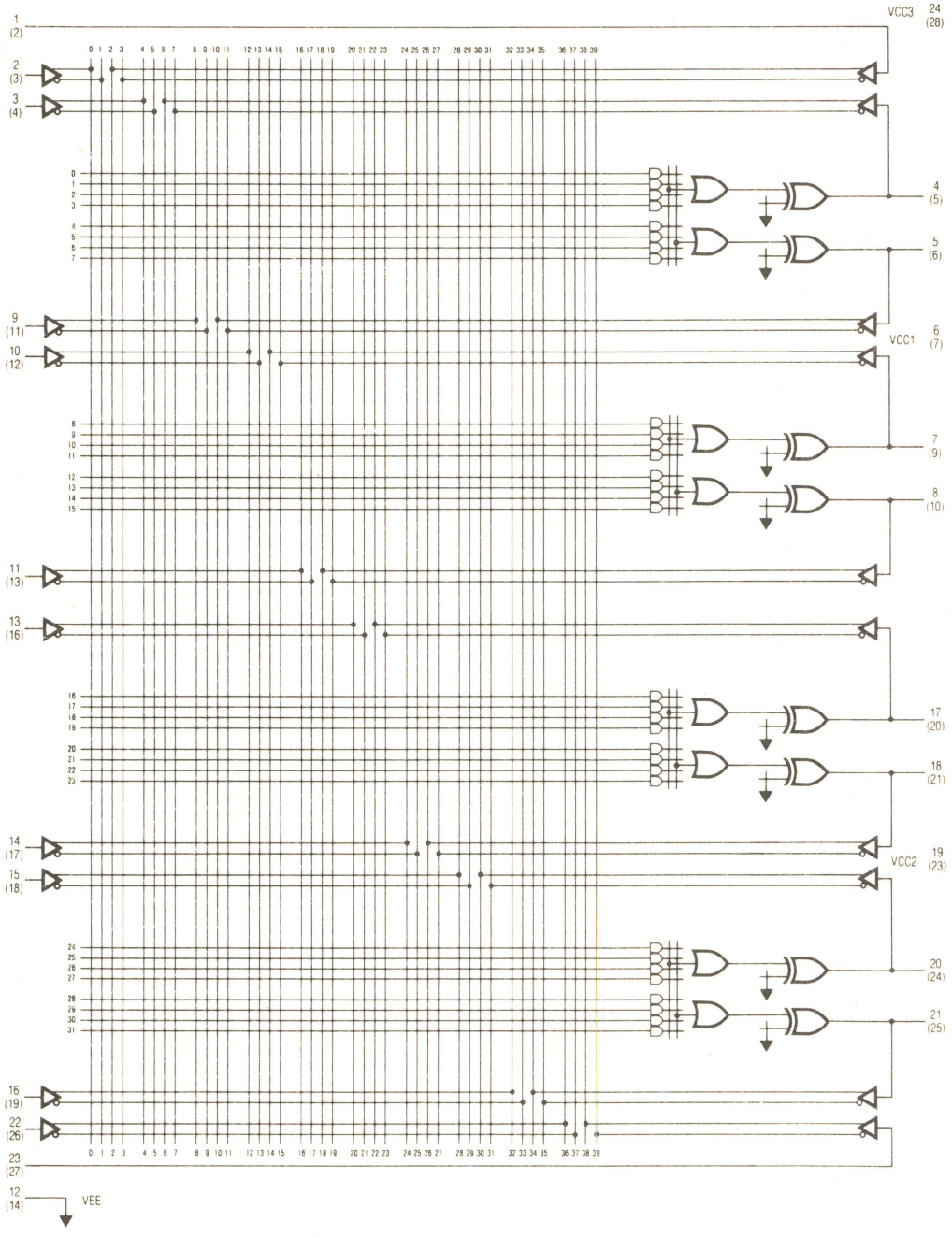
## Switching Characteristics $V_{EE}$ = -5.2 V ±5% (See note 2)

| SYMBOL | PARAMETER | 0°C MIN | 0°C MAX | 25°C MIN | 25°C MAX | 75°C MIN | 75°C MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Propagation delay | 2.0 | 6.0 | 2.0 | 6.0 | 2.0 | 6.0 | ns |
| $t_R$ | Rise time (20%-80%) | 0.7 | 2.2 | 0.7 | 2.0 | 0.7 | 2.2 | ns |
| $t_F$ | Fall time (80%-20%) | 0.7 | 2.2 | 0.7 | 2.0 | 0.7 | 2.2 | ns |

Notes:

1. Each ECL 10KH series circuit has been designed to meet the specifications shown in test table after thermal equilibrium has been established. The circuit is in test socket or mounted on a printed board and transverse air flow greater than 500 linear fpm is maintained.
2. Outputs are terminated through a 50 Ω resistor to $V_{CC}$ -2.0 V. Discrete carbon resistors should be used for terminations. Multiple-resistor packs and metal film discrete resistors are inductive and should be avoided. The single-ended nature of the outputs demands strict adherence to ground and termination plane design techniques.
3. If pin 13 (PLCC pin 16) is not used, it should be left open or terminated to $V_{TT}$ (= $V_{CC}$ -2.0 V). It should not be terminated to $V_{EE}$.

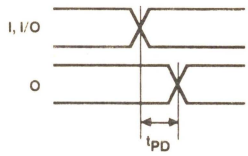
## Logic Diagram




Note: Numbers in parentheses refer to the PLCC pin number.
PLCC pins 1, 8, 15, and 22 are not connected.

## Definitions of Switching Parameters

$t_{PD}$: Signal propagation delay from an input or an I/O pin through the array to a combinatorial output or a latched output while G is low.



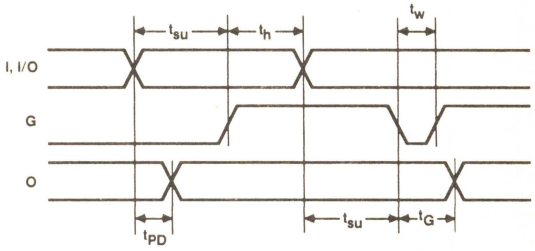

$t_{su}$: Time that input data must be valid before G goes high in order to latch data

$t_h$: Time that input data must be valid after G goes high in order to latch data

$t_W$: The minimum gate low pulse width needed to latch new data

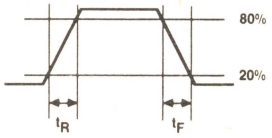
$t_G$: Delay between lowering G and data appearing at the output.

Note: In order for a signal to appear at the output at a time $t_G$ after lowering G, the signal must be setup a time $t_{su}$ before G is lowered. If this amount of time is not allowed, then the output will change at a time $t_{PD}$ after the inputs were changed. The $t_{su}$ needed is illustrated in the waveforms.



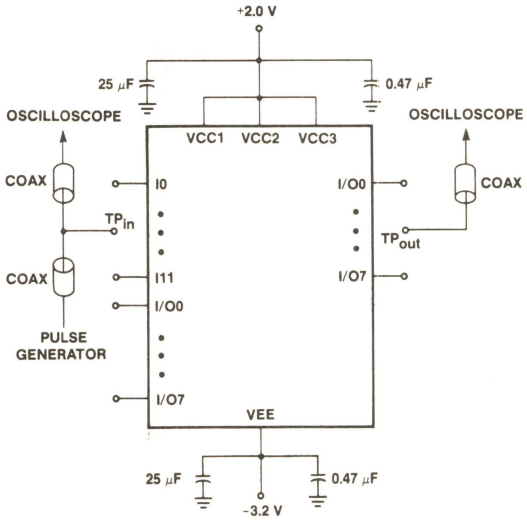

$t_R$: Time taken for an output signal voltage to swing from 20% to 80% of the full logic swing

$t_F$: Time taken for an output signal voltage to swing from 80% to 20% of the full logic swing




5

## Setup for Testing Switching Characteristics




Each oscilloscope channel input should have a 50 Ω termination to ground. Oscilloscope bandwidth should be at least 1 GHz.

The pulse generator should be capable of providing 1.5 ns rise and fall times (20% to 80%).

All input and output cables should be equal lengths of matched 50 Ω coaxial cable. Wire lengths between input (or I/O) pins and $TP_{in}$ or between output pins and $TP_{out}$ should be less than 1/4 in. long. Stubs should be avoided if possible; unavoidable stubs should be less than 2 in. long.

Used inputs that are not switching should be forced to $V_{IL}$ or $V_{IH}$.

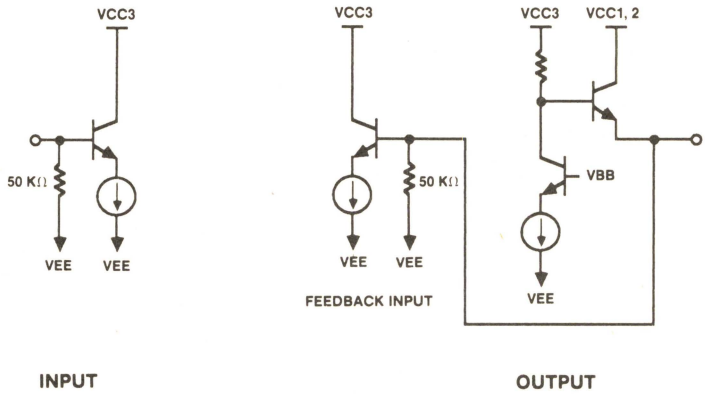
Outputs that are switching but not sensed should be terminated through 50Ω to ground.

Unused inputs and outputs may be left open. If unused pins are to be terminated, pin 14 (PLCC pin 17) should be terminated to $V_{TT}$ (= $V_{CC}$ -2.0 V) and not to $V_{EE}$.

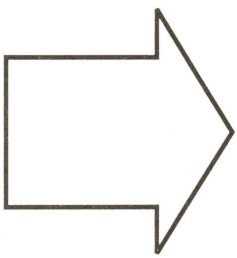
Note that all voltages are shifted by +2.0 V with respect to normal ECL operating conditions in order to take advantage of the input terminations of the oscilloscope.

Timing thresholds in this configuration are taken to be +0.7 V.

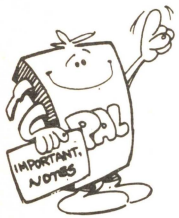
## Input and Output Equivalent Schematics



## Data Sheets



- TTL/CMOS PAL Devices
- TTL/CMOS AmPAL Devices
- PROSE/PLS Sequencers
- FPC/PEG Sequencers
- ECL PAL Devices
- HAL/ZHAL Devices
- Military PAL Devices
- Logic Cell Array
- Electrical Definitions


IMPORTANT
NOTES

# HAL/ZHAL Devices

ProPAL, HAL and ZHAL devices are programmable logic devices that are programmed, marked and functionally tested by Monolithic Memories. Our functional testing offers the user board-ready product at quality levels as stringent as 50 Parts Per Million (PPM), providing significant benefits in both quality and manufacturing cost savings. The ProPAL, HAL and ZHAL device program provides system manufacturers a risk-free migration path from system prototype to full production with extremely high-quality, board-ready devices.

## ProPAL Devices

ProPAL (Programmed PAL) devices are simply PAL devices that Monolithic Memories programs and tests for you. You receive a fully functional device without having to do any programming and testing, and still have the flexibility to handle design changes easily.

## HAL Devices

HAL (Hard Array Logic) devices are to PAL devices as ROMs are to PROMs. Instead of fuses in the logic array, your pattern is implemented using metal links that are masked in during wafer fabrication.

## ZHAL Devices

ZHAL devices are Zero-Standby-Power CMOS HAL devices. These devices can implement any pattern from our standard and combinatorial 20-pin and 24-pin PAL device families with the greatly reduced power consumption only CMOS can offer.

All ZHAL devices are fully HC/HCT compatible, making them easy to use in TTL and CMOS environments.

For a complete discussion of the benefits of ProPAL, HAL, and ZHAL devices, see page 3-104.

### Device Availability

The HAL device option is available for most products, including the following:

ZHAL16RP8A Series (CMOS ZHAL20A Series)

HAL16R8D Series
HAL16R8B Series
HAL16R8B-2 Series
HAL16R8A Series
HAL16R8B-4 Series
HAL16R8A-2 Series
HAL16R8A-4 Series
ZHAL16R8A Series (CMOS ZHAL20A Series)

HAL10H8A Series
ZHAL10H8A Series (CMOS ZHAL20A Series)

HAL32VX10/A

ZHAL20RS10A Series (CMOS ZHAL24A Series)

HAL20X10A Series
ZHAL20X10A Series (CMOS ZHAL24A Series)

HAL20R8B Series
HAL20R8A Series
HAL20R8A-2 Series
ZHAL20R8A Series (CMOS ZHAL24A Series)

HAL12L10A Series
ZHAL12L10A Series (CMOS ZHAL24A Series)

HAL32R16

All HAL device specifications are equivalent to the corresponding PAL device specifications. Two exceptions are the HAL10H8A and HAL12L10A combinatorial series at 25 ns, which have no equivalent PAL device option. The CMOS ZHAL device specifications follow.

To convert a PAL device design to a ProPAL, HAL, or ZHAL device, contact your local sales office.

# Zero Power CMOS Hard Array Logic ZHAL™ 20A Series

## Features/Benefits

- **Zero standby power**
- **25-ns maximum propagation delay**
- **HC and HCT compatible**
- **Space saving PLCC available**
- **Low power alternative for Small and Medium 20-pin PAL® devices, including 16L8/16R8/16R6/16R4**

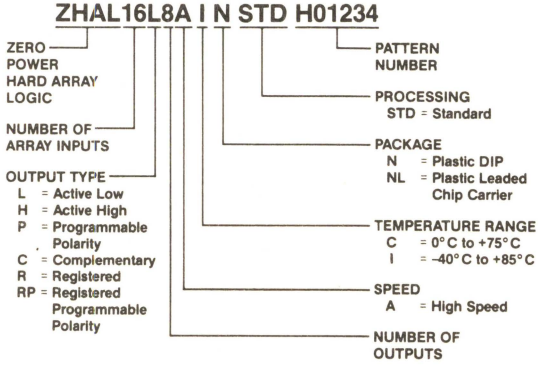
## Description

The Zero Power Hard Array Logic (ZHAL) devices are ideal in low-power applications that require high-speed operation. These attributes are achieved through the use of Monolithic Memories' advanced high-speed CMOS process. Now system designers have the option of using a ZHAL device that matches fast PAL device speeds, but with the added advantage of zero standby power. These features are ideal for power-critical areas such as portable digital equipment or lap-top computers.

This family of ZHAL devices utilizes a unique architecture that is designed for a high degree of flexibility in implementing most patterns of the listed 20-pin PAL/HAL® devices. Prototyping can be done using standard PAL devices before converting to ZHAL circuits for production. ZHAL devices are fabricated by Monolithic Memories with custom metallization masks defined by a user-supplied HAL Design Specification.

## Ordering Information

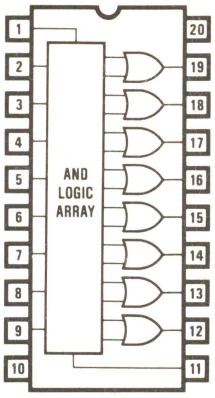
| PART NUMBER | PACKAGE | ARRAY | OUTPUTS | |
|---|---|---|---|---|
| | | | COMB | REG |
| ZHAL10H8A | N, NL | 10 | 8 | — |
| ZHAL12H6A | | 12 | 6 | — |
| ZHAL14H4A | | 14 | 4 | — |
| ZHAL16H2A | | 16 | 2 | — |
| ZHAL16C1A | | 16 | 2 | — |
| ZHAL10L8A | | 10 | 8 | — |
| ZHAL12L6A | | 12 | 6 | — |
| ZHAL14L4A | | 14 | 4 | — |
| ZHAL16L2A | | 16 | 2 | — |
| ZHAL16L8A | N, NL | 16 | 8 | — |
| ZHAL16R8A | | 16 | — | 8 |
| ZHAL16R6A | | 16 | 2 | 6 |
| ZHAL16R4A | | 16 | 4 | 4 |
| ZHAL16P8A | N, NL | 16 | 8 | — |
| ZHAL16RP8A | | 16 | — | 8 |
| ZHAL16RP6A | | 16 | 2 | 6 |
| ZHAL16RP4A | | 16 | 4 | 4 |



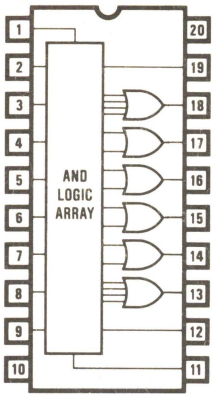

PAL®, HAL®, and PALASM® are registered trademarks of Monolithic Memories.
ZHAL™ and ProPAL™ are trademarks of Monolithic Memories.

10237A
JANUARY 1988

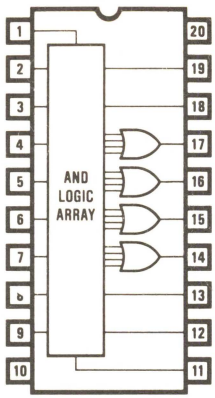
ZHAL10H8A



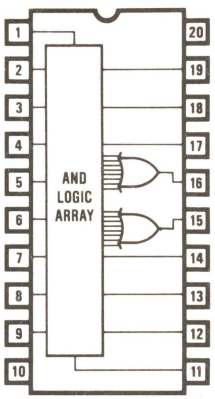

ZHAL12H6A



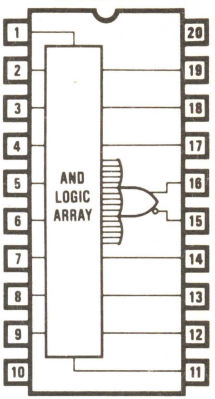

ZHAL14H4A



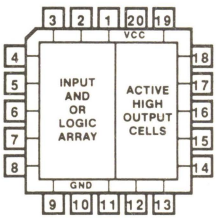

ZHAL16H2A




ZHAL16C1A

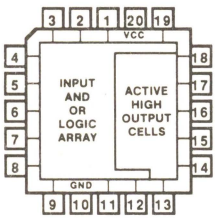



ZHAL10H8A



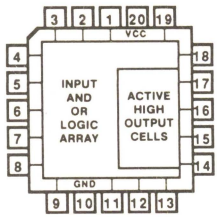

Plastic Chip Carrier

ZHAL12H6A



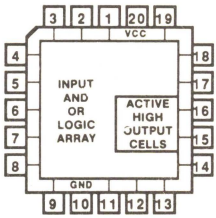

Plastic Chip Carrier

ZHAL14H4A



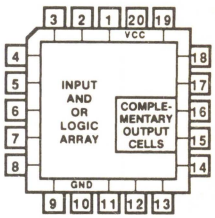

Plastic Chip Carrier

ZHAL16H2A



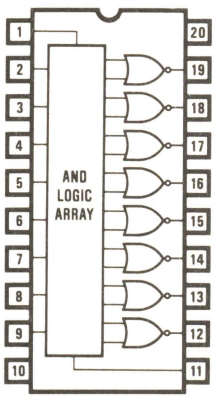

Plastic Chip Carrier

ZHAL16C1A

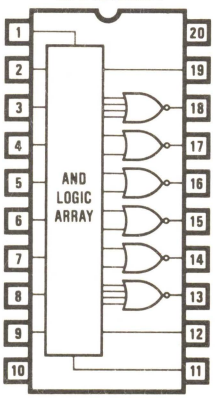



Plastic Chip Carrier

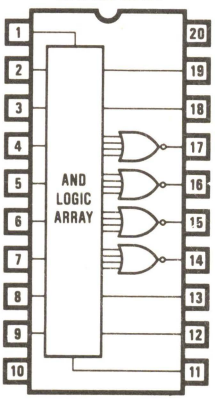
ZHAL10L8A



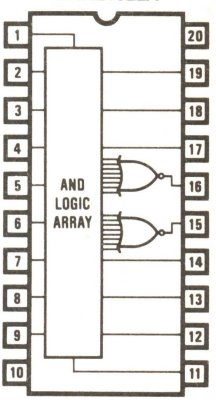

ZHAL12L6A



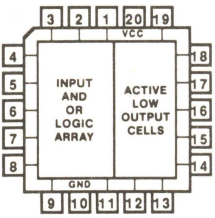

ZHAL14L4A




ZHAL16L2A

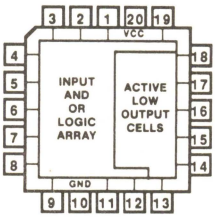



ZHAL10L8A



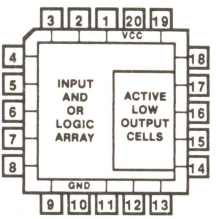

Plastic Chip Carrier

ZHAL12L6A



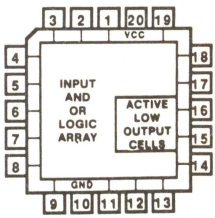

Plastic Chip Carrier

ZHAL14L4A




Plastic Chip Carrier

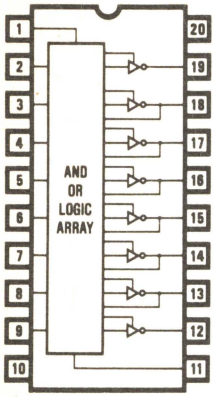
ZHAL16L2A



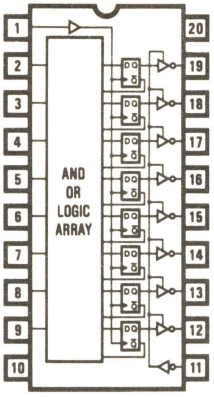

Plastic Chip Carrier

5

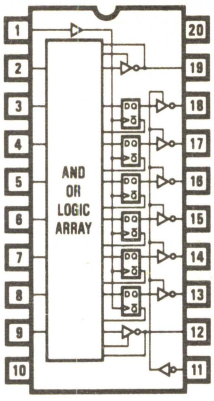
ZHAL16L8A



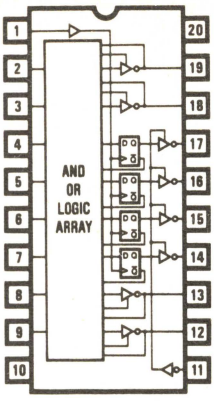

ZHAL16R8A



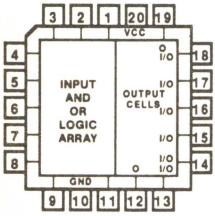

ZHAL16R6A




ZHAL16R4A

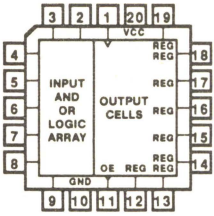



ZHAL16L8A



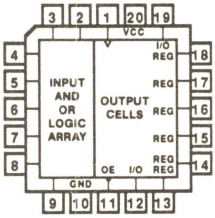

Plastic Chip Carrier

ZHAL16R8A



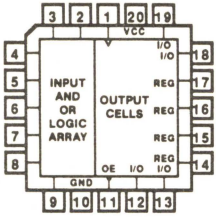

Plastic Chip Carrier

ZHAL16R6A



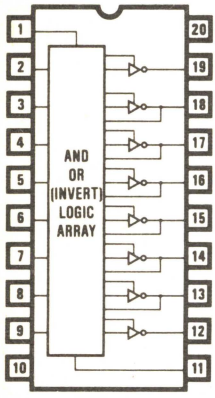

Plastic Chip Carrier

ZHAL16R4A

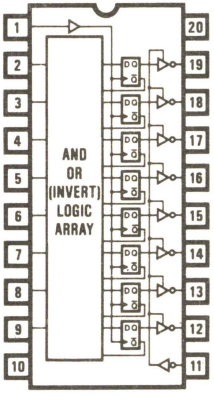



Plastic Chip Carrier

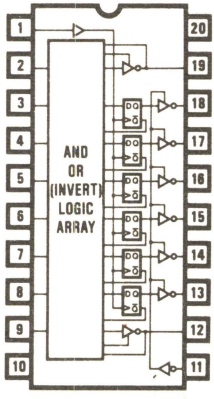
ZHAL16P8A



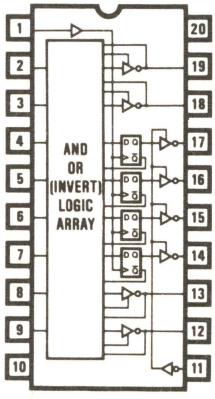

ZHAL16RP8A



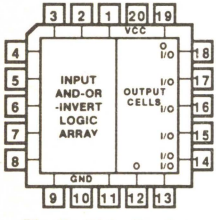

ZHAL16RP6A




ZHAL16RP4A

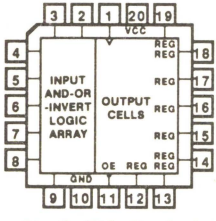



ZHAL16P8A



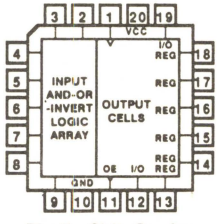

Plastic Chip Carrier

ZHAL16RP8A



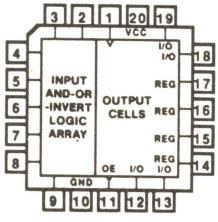

Plastic Chip Carrier

ZHAL16RP6A




Plastic Chip Carrier

ZHAL16RP4A




Plastic Chip Carrier

## Operating Conditions

| SYMBOL | PARAMETER | | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | 16R4A, 16R6A, 16R8A, 16RP4A, 16RP6A, 16RP8A | 15 | 10 | | 15 | 10 | | ns |
| $t_{su}$ | Setup time from input or feedback to clock | | 20 | 13 | | 20 | 13 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | −40 | 25 | 85 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | | 0 | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | | $V_{CC}$ = MAX | $V_I$ = GND | | | −1 | μA |
| $I_{IH}$ | High-level input current | Pin 8[2] | $V_{CC}$ = MAX | $V_I$ = $V_{CC}$ | | 1 | 10 | μA |
| | | All other pins | | | | | 1 | μA |
| $V_{OL}$ | Low-level output voltage | | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.1 | 0.4 | V |
| | | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 μA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | | $V_{CC}$ = MIN | $I_{OH}$ = −6 mA | 3.76[3] | 4.1 | | V |
| | | | $V_{CC}$ = 5 V | $I_{OH}$ = −1 μA | 4.95 | | | |
| $I_{OZL}$[3] | Off-state output current | | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | −10 | μA |
| $I_{OZH}$[3] | | | | $V_O$ = $V_{CC}$ | | 0 | 10 | μA |
| $I_{CC}$ | Standby supply current[4] | | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | μA |
| | Operating supply current | | f = 1 MHz, $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 2 | 5[5] | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS (See Test Load) | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 10H8A, 12H6A, 14H4A, 16H2A, 16C1A, 10L8A, 12L6A, 14L4A 16L2A, 16L8A, 16R6A, 16R4A, 16P8A, 16RP6A, 16RP8A | | $R_L$ = 1 KΩ $C_L$ = 50 pF | | 15 | 25 | | 15 | 25 | ns |
| $t_{CLK}$ | Clock to output or feedback 16R4A, 16R6A, 16R8A, 16RP4A, 16RP6A, 16RP8A | | | | 10 | 15 | | 10 | 15 | ns |
| $t_{PZX}$ | Input to output enable | 16L8A, 16R4A, 16R6A, 16P8A, 16RP4A, 16RP6A | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[6] | Input to output disable | | | | 14 | 25 | | 14 | 25 | ns |
| $t_{PXZ}$[6] $t_{PZX}$ | Pin 11 to output disable/enable | 16R4A, 16R6A, 16R8A, 16RP4A, 16RP6A, 16RP8A | | | 12 | 15 | | 12 | 15 | ns |
| $f_{MAX}$ | Maximum frequency | | | 28.5 | 40 | | 28.5 | 40 | | MHz |

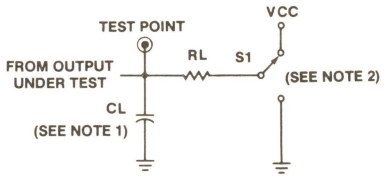
Notes: Apply to electrical and switching characteristics.

1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. Pin 8 (PRELOAD pin). Applies to all devices whether registered or non-registered.
3. JEDEC standard no. 7 for high-speed CMOS devices.
4. Disable output pins = $V_{CC}$ or GND.
5. Add 3 mA per additional 1.0 MHz of operation over 1 MHz.
6. $C_L$ = 5 pF.

## Absolute Maximum Ratings

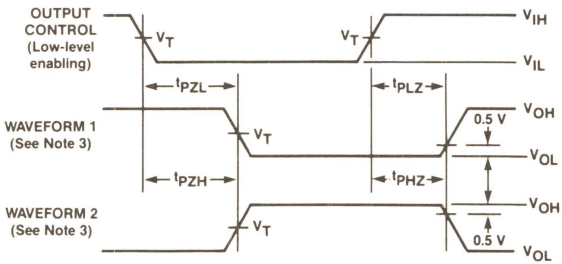
Supply voltage, $V_{CC}$ ..... -0.5 V to 7 V
DC input voltage, $V_I$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output voltage, $V_O$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output source/sink current per output pin, $I_O$ ..... ±35 mA
DC $V_{CC}$ or ground current, $I_{CC}$ or $I_{GND}$ ..... ±100 mA
Input diode current, $I_{IK}$:
$V_I$<0 ..... -20 mA
$V_I$>$V_{CC}$ ..... +20 mA
Output diode current, $I_{OK}$:
$V_O$<0 ..... -20 mA
$V_O$>$V_{CC}$ ..... +20 mA
Storage temperature ..... -65°C to 150°C

## Switching Test Load

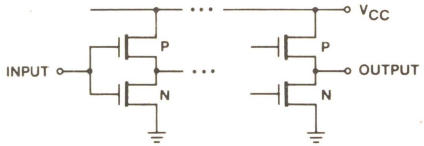



Notes: 1. CL includes probe and jig capacitance.
2. When measuring $t_{PLZ}$ and $t_{PZL}$, S1 is tied to $V_{CC}$.
When measuring $t_{PHZ}$ and $t_{PZH}$, S1 is tied to ground.
$t_{PZX}$ is measured with $C_L$ = 50 pF. $t_{PXZ}$ is measured with $C_L$ = 5 pF.
When measuring propagation delay times of 3-state outputs, S1 is open, i.e., not connected to $V_{CC}$ or ground.
3. Waveform 1 is for an output with internal conditions such that the output is Low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is High except when disabled by the output control.

## Enable/Disable Delay




## Schematic of Inputs and Outputs

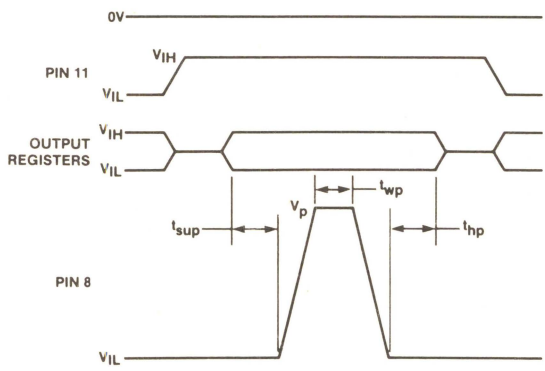



## Output Register PRELOAD†

The PRELOAD function allows the register to be loaded from data placed on the output pins. This feature aids functional testing of state sequencer designs by allowing direct setting of output states for improved test coverage. The procedure for PRELOAD is as follows:

1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 11 to $V_{IH}$. Set pin 1 to 0 V.
3. Apply $V_{IL}$/$V_{IH}$ to all registered output pins.
4. Pulse pin 8 to $V_p$ (12 V), then back to 0 V.
5. Remove $V_{IL}$/$V_{IH}$ from all registered output pins.
6. Lower pin 11 to $V_{IL}$ to enable the output registers.
7. Verify for $V_{OL}$/$V_{OH}$ at all registered output pins.

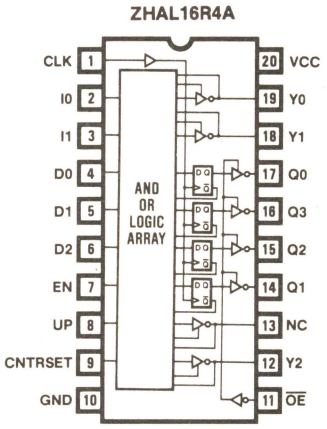
† Note: Only applies to parts with output registers.
Typical $t_{sup}$ = 50 ns
$t_{wp}$ = 100 ns
$t_{hp}$ = 50 ns
$I_{IH}$ = 30 μA (Pin 8)



## Features/Benefits

- Demonstrates all features of ZHAL20A product
- 4-bit up/down counter with reset
- 3-bit shifter
- 25-ns maximum propagation delay
- Zero standby power

## Logic Symbol




## Description

The ZHAL20A Evaluation Pattern is provided as an example of the features and characteristics of the ZHAL20A Series products.

This design consists of two functionally independent patterns: a 4-bit up/down counter and a 3-bit shifter. The 4-bit counter can count up or count down and has reset capability. These features are controlled by two control signals: UP and CNTRSET (Count Reset). When UP is high, the counter counts up. When UP is low, the counter counts down. CNTRSET overrides the count function and resets the counter to all ones, synchronous with the clock.

The 3-bit shifter shifts data bits by 0, 1 or 2 positions. The three bits of the shifter are enabled when EN (enable) is high, and are disabled (high-Z) when EN is low.

The PALASM®2 software file and simulation results are shown on the next page. Below are the function tables that summarize the functions of the counter and the shifter.

## Counter Function Table

| $\overline{OE}$ | UP | CNTRSET | CLK | Q3-Q0 | OPERATION |
|---|---|---|---|---|---|
| H | X | X | X | Z | High-Z |
| L | H | L | ↑ | Q plus 1 | Increment |
| L | L | L | ↑ | Q minus 1 | Decrement |
| L | X | H | ↑ | High | Reset |

H = HIGH voltage level
L = LOW voltage level
X = Don't care
Z = High impedance (off) state
↑ = LOW-to-HIGH clock transition

## Shifter Function Table

| EN | I1 | I0 | Y2 | Y1 | Y0 | OPERATION |
|---|---|---|---|---|---|---|
| L | X | X | Z | Z | Z | High-Z |
| H | L | L | $\overline{D2}$ | $\overline{D1}$ | $\overline{D0}$ | No operation |
| H | L | H | $\overline{D0}$ | $\overline{D2}$ | $\overline{D1}$ | Shift by one |
| H | H | L | $\overline{D1}$ | $\overline{D0}$ | $\overline{D2}$ | Shift by two |

5

## PALASM Design Specification

```
TITLE     PDS CONVERSION FILE
PATTERN   EXAMPLE
REVISION  1.00
AUTHOR    JOHN DOE
COMPANY   MONOLITHIC MEMORIES, INC
DATE      9/23/85

CHIP zzz PAL16RP4 CLK I0 I1 D0 D1 D2 EN UP CNTRSET GND
                  /OE Y2 NC Q1 Q2 Q3 Q0 Y1 Y0 VCC

EQUATIONS

 Y0 = /I1*/I0*/D0
    + /I1* I0*/D1
    +  I1*/I0*/D2
 Y0.TRST = EN

 Y1 = /I1*/I0*/D1
    + /I1* I0*/D2
    +  I1*/I0*/D0
 Y1.TRST = EN

 Y2 = /I1*/I0*/D2
    + /I1* I0*/D0
    +  I1*/I0*/D1
 Y2.TRST = EN


Q0 :=/Q0
    + CNTRSET

Q1 :=/Q1* Q0* UP
    + Q1* Q0*/UP
    + Q1*/Q0* UP
    +/Q1*/Q0*/UP
    + CNTRSET

Q2 := CNTRSET
    +/Q2*/Q1*/Q0*/UP
    + Q2*/Q1*     UP
    + Q2* Q1*/Q0
    +/Q2* Q1* Q0* UP
    + Q2*     Q0*/UP

Q3 := CNTRSET
    + Q3*          Q0*/UP
    +/Q3*/Q2*/Q1*/Q0*/UP
    +/Q3* Q2* Q1* Q0* UP
    + Q3* Q2*/Q1
    + Q3*    /Q1*     UP
    + Q3*/Q2* Q1
    + Q3*     Q1*/Q0
```

## Simulation Results

```
PALASM SIMULATION HISTORY LISTING

ZZZ
Page :  1
          g g  cg cg  cg cgcgcg cgcgcgcgcg cgcgcgcgc
 CLK      LLLLHLLHLL HLLHLHLHLH LHLHLHLHLH LHLHLHLHL
 I0       XXLLLLLLLH HHLLLLLLLL LLLLLLLLLL LLLLLLLLL
 I1       XXLLLLLLL LLHHHHHHHH HHHHHHHHHH HHHHHHHHH
 D0       XXLLLLLLLH HHLLLLLLLL LLLLLLLLLL LLLLLLLLL
 D1       XXLLLLLLLL LLHHHHHHHH HHHHHHHHHH HHHHHHHHH
 D2       XXLLLLLLLH HHLLLLLLLL LLLLLLLLLL LLLLLLLLL
 EN       LLHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHH
 UP       XXXXXXHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHH
 CNTRSET  XXHHHHLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLL
 GND      LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLL
/OE       HHLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLL
 Y2       XZHHHHHHHL LLLLLLLLLL LLLLLLLLLL LLLLLLLLL
 Q1       ZZXXXHHHLL LLLLHHHHLL LLHHHHLLLL HHHHLLLLH
 Q2       ZZXXXHHHLL LLLLLLLLHH HHHHHHLLLL LLLLHHHHH
 Q3       ZZXXXHHHLL LLLLLLLLLL LLLLLLHHHH HHHHHHHHH
 Q0       ZZXXXHHHLL LHHHLLHHLL HHLLHHLLHH LLHHLLHHL
 Y1       XZHHHHHHHL LLHHHHHHHH HHHHHHHHHH HHHHHHHHH
 Y0       XZHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHH
 VCC      HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHH
```

## Simulation File

```
SIMULATION

TRACE_ON CLK I0 I1 D0 D1 D2 EN UP CNTRSET /OE Y2 Q1 Q2 Q3 Q0 Y1 Y0

SETF /CLK /OE /EN

SETF OE EN /I1 /I0 /D2 /D1 /D0 Y2 Y1 Y0 CNTRSET
CLOCKF CLK

CHECK Q3 Q2 Q1 Q0

SETF /I1 /I0 /D2 /D1 /D0 Y2 Y1 Y0 /CNTRSET UP
CLOCKF CLK

CHECK /Q3 /Q2 /Q1 /Q0

SETF /I1 I0 D2 /D1 D0 /Y2 /Y1 Y0
CLOCKF CLK

CHECK /Q3 /Q2 /Q1 Q0

SETF I1 /I0 /D2 D1 /D0 /Y2 Y1 Y0
CLOCKF CLK

CHECK /Q3 /Q2 Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK /Q3 /Q2 Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK /Q3 Q2 /Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK /Q3 Q2 /Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK /Q3 Q2 Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK /Q3 Q2 Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 /Q2 /Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 /Q2 /Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 /Q2 Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 /Q2 Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 Q2 /Q1 /Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 Q2 /Q1 Q0

SETF OE /CNTRSET UP
CLOCKF CLK

CHECK Q3 Q2 Q1 /Q0

TRACE_OFF
```

# Zero Power CMOS Hard Array Logic ZHAL™24A Series

## Features/Benefits

- **Zero standby power**
- **Low power operation**
- **High-speed CMOS technology**
- **HC and HCT compatible**
- **24-pin SKINNYDIP® and 28-pin PLCC packages save space**
- **Low power alternative for most 24-pin PAL® devices, including 20L8/ 20R8/ 20R6/ 20R4**

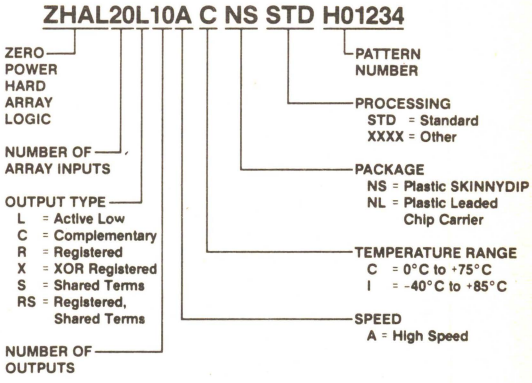
## Description

This family of Zero Power Hard Array Logic (ZHAL) devices utilizes a unique architecture that is designed for a high degree of flexibility in implementing most patterns of the listed 24-pin PAL/HAL® devices. Prototyping should be done using standard PAL devices before converting to ZHAL circuits for production. ZHAL devices are fabricated by Monolithic Memories with custom metallization masks defined by a user-supplied HAL Design Specification.

The ZHAL devices are ideal in low-power applications that require high-speed operation. These attributes are achieved through the use of Monolithic Memories' advanced high-speed CMOS process. Now system designers have the option of using a ZHAL device that matches fast PAL device speeds, but with the added feature of zero standby power. These features are needed in power-critical areas such as portable digital equipment or lap-top computers.

## Ordering Information

| PART NUMBER | PACKAGE | ARRAY INPUTS | OUTPUTS | |
|---|---|---|---|---|
| | | | COMB | REG |
| ZHAL12L10A | NS, NL | 12 | 10 | — |
| ZHAL14L8A | NS, NL | 14 | 8 | — |
| ZHAL16L6A | NS, NL | 16 | 6 | — |
| ZHAL18L4A | NS, NL | 18 | 4 | — |
| ZHAL20L2A | NS, NL | 20 | 2 | — |
| ZHAL20C1A | NS, NL | 20 | 2 | — |
| ZHAL20L8A | NS, NL | 20 | 8 | — |
| ZHAL20R8A | NS, NL | 20 | — | 8 |
| ZHAL20R6A | NS, NL | 20 | 2 | 6 |
| ZHAL20R4A | NS, NL | 20 | 4 | 4 |
| ZHAL20L10A | NS, NL | 20 | 10 | — |
| ZHAL20X10A | NS, NL | 20 | — | 10 |
| ZHAL20X8A | NS, NL | 20 | 2 | 8 |
| ZHAL20X4A | NS, NL | 20 | 6 | 4 |
| ZHAL20S10A | NS, NL | 20 | 10 | — |
| ZHAL20RS10A | NS, NL | 20 | — | 10 |
| ZHAL20RS8A | NS, NL | 20 | 2 | 8 |
| ZHAL20RS4A | NS, NL | 20 | 6 | 4 |



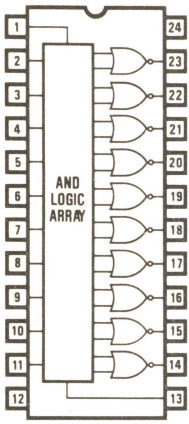

5

PAL®, HAL®, SKINNYDIP® and PALASM® are registered trademarks of Monolithic Memories.
ZHAL™ and ProPAL™ are trademarks of Monolithic Memories.

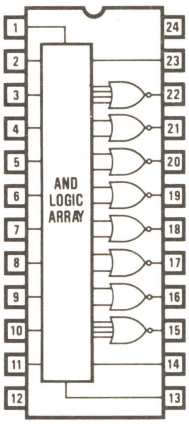
10242A
JANUARY 1988

## Pin Configurations — DIP and PLCC

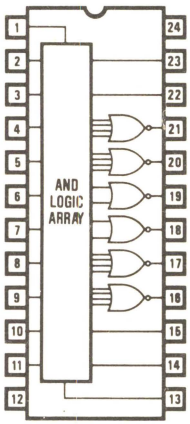
ZHAL12L10A



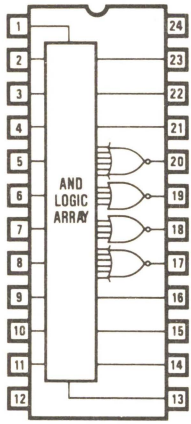

ZHAL14L8A



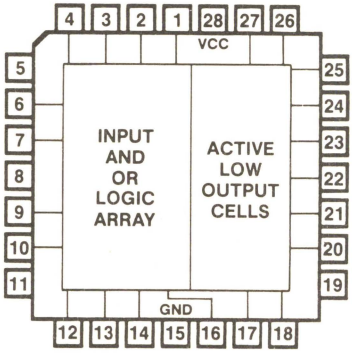

ZHAL16L6A



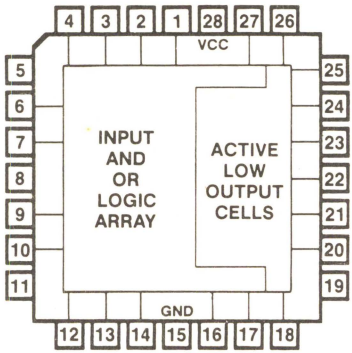

ZHAL18L4A



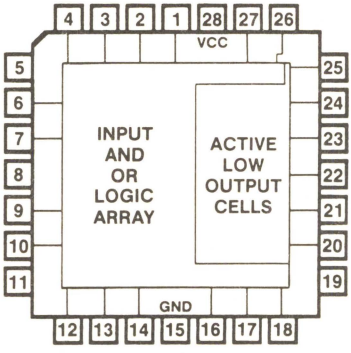

ZHAL12L10A



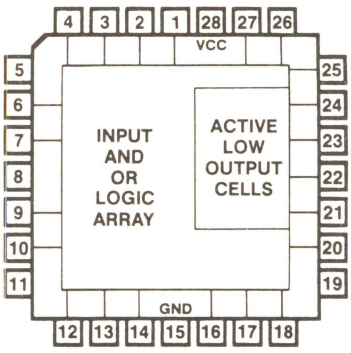

ZHAL14L8A




ZHAL16L6A

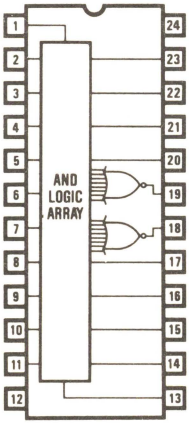



ZHAL18L4A

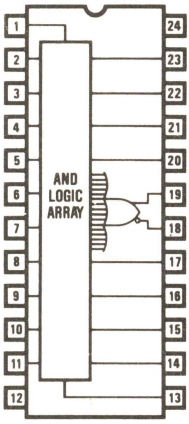


## Pin Configurations — DIP and PLCC

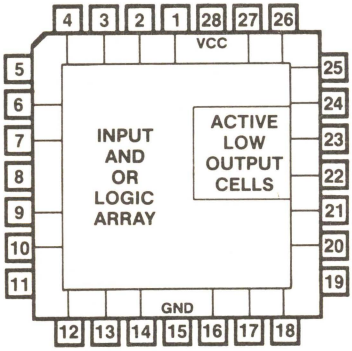
ZHAL20L2A



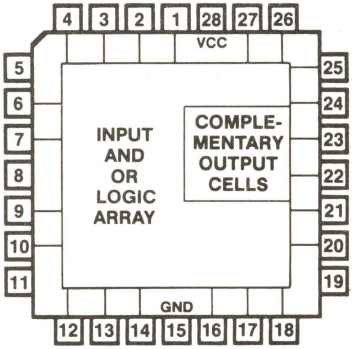

ZHAL20C1A




ZHAL20L2A



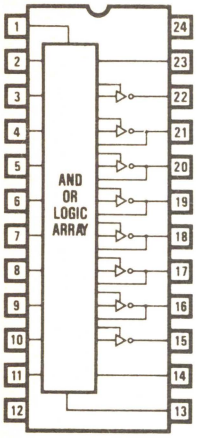

ZHAL20C1A



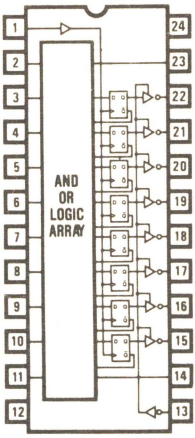

5

## Pin Configurations — DIP and PLCC

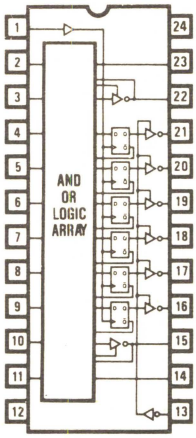
ZHAL20L8A



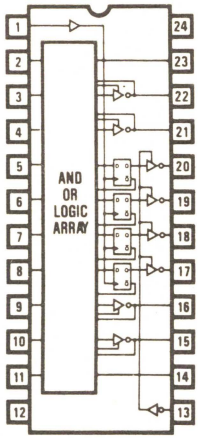

ZHAL20R8A



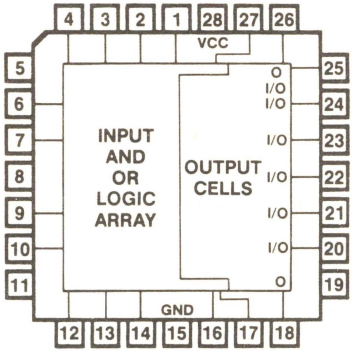

ZHAL20R6A



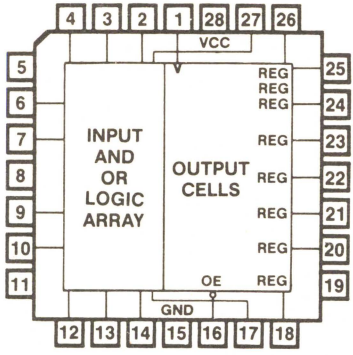

ZHAL20R4A



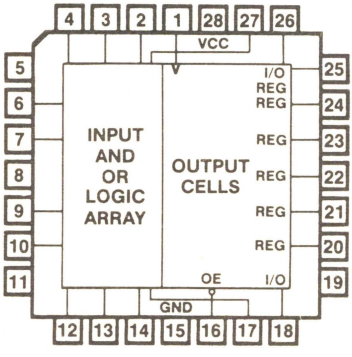

ZHAL20L8A



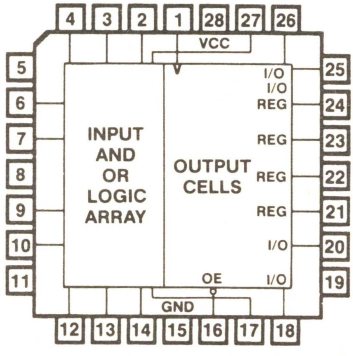

ZHAL20R8A




ZHAL20R6A

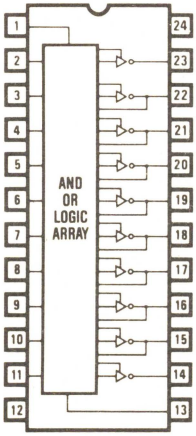



ZHAL20R4A

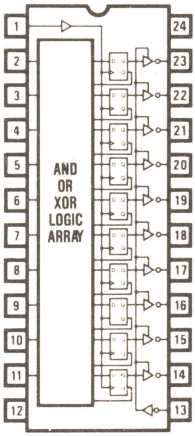


## Pin Configurations — DIP and PLCC

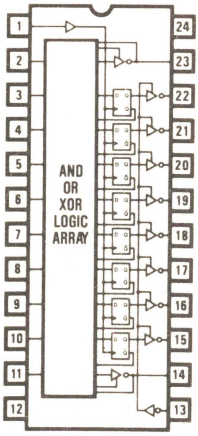
ZHAL20L10A



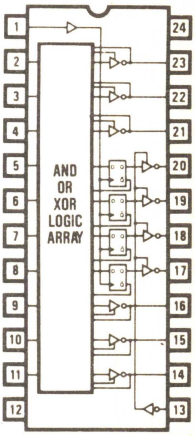

ZHAL20X10A



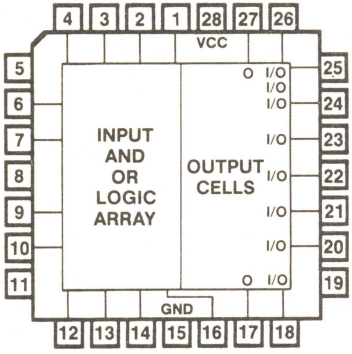

ZHAL20X8A



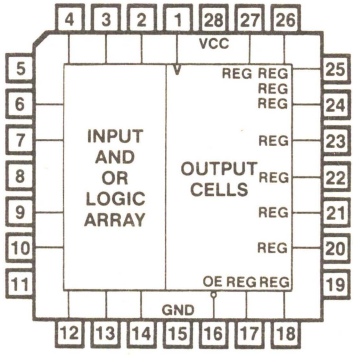

ZHAL20X4A



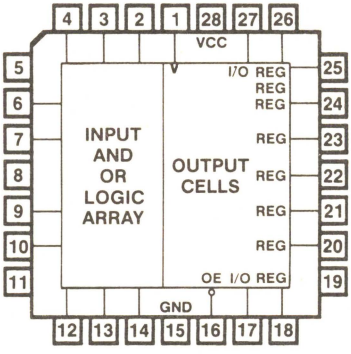

ZHAL20L10A



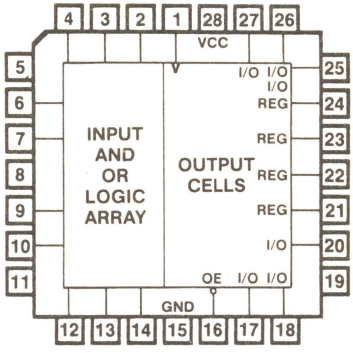

ZHAL20X10A




ZHAL20X8A



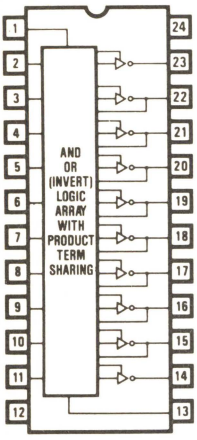

ZHAL20X4A



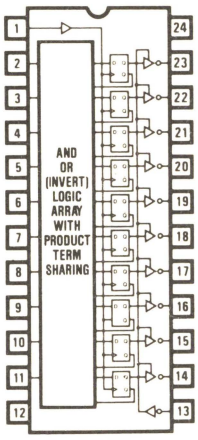

5

## Pin Configurations — DIP and PLCC

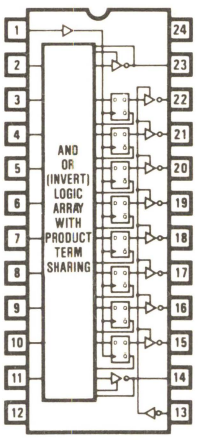
ZHAL20S10A



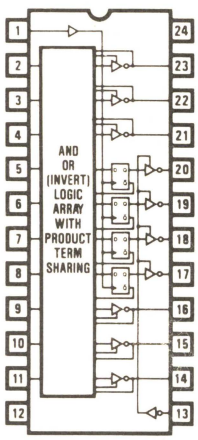

ZHAL20RS10A



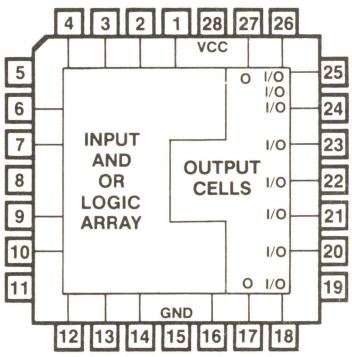

ZHAL20RS8A



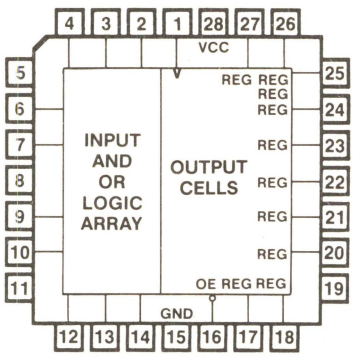

ZHAL20RS4A



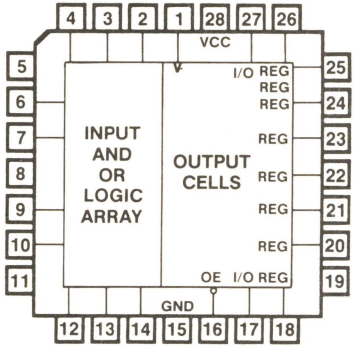

ZHAL20S10A



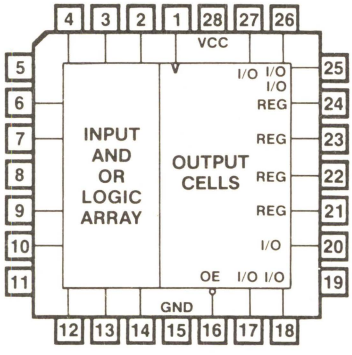

ZHAL20RS10A




ZHAL20RS8A




ZHAL20RS4A



## Operating Conditions

| SYMBOL | PARAMETER | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | V |
| $T_A$ | Operating free-air temperature | -40 | 25 | 85 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IL}$[1] | Low-level input voltage | | | 0 | | 0.8 | V |
| $V_{IH}$[1] | High-level input voltage | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX $V_I$ = GND | | | | -1 | μA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX $V_I$ = $V_{CC}$ | | | | 1 | μA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.1 | 0.4 | V |
| | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 μA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = -6 mA | 3.76[2] | 4.1 | | V |
| | | $V_{CC}$ = 5 V | $I_{OH}$ = -1 μA | 4.95 | | | |
| $I_{OZL}$[3] | Off-state output current | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | -10 | μA |
| $I_{OZH}$[3] | | | $V_O$ = $V_{CC}$ | | 0 | 10 | μA |
| $I_{CC}$ | Standby supply current[4] | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | μA |
| | Operating supply current | f = 1 MHz, $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 2 | 5[5] | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS (See Test Load) | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input to output | $R_L$ = 1 KΩ<br>$C_L$ = 50 pF | | 13 | 25[6] | | 13 | 25[6] | ns |

Notes:
1. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
2. JEDEC standard no. 7 for high-speed CMOS devices.
3. Applies to pins 14-23 for DIP (pins 17, 18, 20-27 for PLCC).
4. Disable output pins = $V_{CC}$ or GND.
5. Add 3 mA per additional 1.0 MHz of operation over 1 MHz.
6. For outputs with more than 12 inputs in a product term, $t_{PD}$ = 30 ns.

## Operating Conditions

| SYMBOL | PARAMETER | | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | 20R8A, 20R6A, 20R4A | 15 | 4 | | 15 | 4 | | ns |
| $t_{su}$ | Setup time from input or feedback to clock | | 20[1] | 11 | | 20[1] | 11 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | -40 | 25 | 85 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | 0 | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | | $V_{CC}$ = MAX | $V_I$ = GND | | | -1 | μA |
| $I_{IH}$ | High-level input current | Pin 10[3] | $V_{CC}$ = MAX | $V_I$ = $V_{CC}$ | | 8 | 30 | μA |
| | | All other pins | | | | | 1 | μA |
| $V_{OL}$ | Low-level output voltage | | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.1 | 0.4 | V |
| | | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 μA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | | $V_{CC}$ = MIN | $I_{OH}$ = -6 mA | 3.76[4] | 4.1 | | V |
| | | | $V_{CC}$ = 5 V | $I_{OH}$ = -1 μA | 4.95 | | | |
| $I_{OZL}$[5] | Off-state output current | | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | -10 | μA |
| $I_{OZH}$[5] | | | | $V_O$ = $V_{CC}$ | | 0 | 10 | μA |
| $I_{CC}$ | Standby supply current[6] | | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | μA |
| | Operating supply current | | f = 1 MHz, $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 2 | 5[7] | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS (See Test Load) | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 20L8A, 20R6A, 20R4A | | $R_L$ = 1 KΩ, $C_L$ = 50 pF | | 13 | 25[1] | | 13 | 25[1] | ns |
| $t_{CLK}$ | Clock to output or feedback 20R8A, 20R6A, 20R4A | | | | 8 | 15 | | 8 | 15 | ns |
| $t_{PZX}$ | Input to output enable | 20L8A, 20R6A, 20R4A | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] | Input to output disable | | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] / $t_{PZX}$ | Pin 13 (DIP) to output disable/enable | 20R8A, 20R6A, 20R4A | | | 10 | 20 | | 10 | 20 | ns |
| $f_{MAX}$ | Maximum frequency | | | 28.5 | 40 | | 28.5 | 40 | | MHz |

Notes:
1. For outputs with more than 12 inputs in a product term, $t_{SU}$ = 25 ns and $t_{PD}$ = 30 ns.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. Pin 10 DIP, pin 13 PLCC (PRELOAD pin). Applies to registered devices only.
4. JEDEC standard no. 7 for high-speed CMOS devices.
5. Applies to pins 14-23 for DIP (pins 17, 18, 20-27 for PLCC).
6. Disable output pins = $V_{CC}$ or GND.
7. Add 3 mA per additional 1.0 MHz of operation over 1 MHz.
8. $C_L$ = 5 pF.

## Operating Conditions

| SYMBOL | PARAMETER | | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | 20X10A, 20X8A, 20X4A | 15 | 5 | | 15 | 5 | | ns |
| $t_{su}$ | Setup time from input or feedback to clock | | 25[1] | 15 | | 25[1] | 15 | | ns |
| $t_h$ | Hold time | | 0 | -10 | | 0 | -10 | | ns |
| $T_A$ | Operating free-air temperature | | -40 | 25 | 85 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | 0 | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | | $V_{CC}$ = MAX | $V_I$ = GND | | | -1 | μA |
| $I_{IH}$ | High-level input current | Pin 10[3] | $V_{CC}$ = MAX | $V_I$ = $V_{CC}$ | | 8 | 30 | μA |
| | | All other pins | | | | | 1 | μA |
| $V_{OL}$ | Low-level output voltage | | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.1 | 0.4 | V |
| | | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 μA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | | $V_{CC}$ = MIN | $I_{OH}$ = -6 mA | 3.76[4] | 4.1 | | V |
| | | | $V_{CC}$ = 5 V | $I_{OH}$ = -1 μA | 4.95 | | | |
| $I_{OZL}$[5] | Off-state output current | | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | -10 | μA |
| $I_{OZH}$[5] | | | | $V_O$ = $V_{CC}$ | | 0 | 10 | μA |
| $I_{CC}$ | Standby supply current[6] | | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | μA |
| | Operating supply current | | f = 1 MHz, $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 2 | 5[7] | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS (See Test Load) | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 20L10A, 20X8A, 20X4A | | $R_L$ = 1 KΩ, $C_L$ = 50 pF | | 13 | 25[1] | | 13 | 25[1] | ns |
| $t_{CLK}$ | Clock to output or feedback 20X10A, 20X8A 20X4A | | | | 10 | 15 | | 10 | 15 | ns |
| $t_{PZX}$ | Input to output enable | 20L10A, 20X8A, 20X4A | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] | Input to output disable | | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] / $t_{PZX}$ | Pin 13 (DIP) to output disable/enable | 20X10A, 20X8A, 20X4A | | | 15 | 20 | | 15 | 20 | ns |
| $f_{MAX}$ | Maximum frequency | | | 22.2 | 32 | | 22.2 | 32 | | MHz |

Notes:
1. For outputs with more than 12 inputs in a product term, $t_{SU}$ = 30 ns and $t_{PD}$ = 30 ns.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. Pin 10 DIP, pin 13 PLCC (PRELOAD pin). Applies to registered devices only.
4. JEDEC standard no. 7 for high-speed CMOS devices.
5. Applies to pins 14-23 for DIP (pins 17, 18, 20-27 for PLCC).
6. Disable output pins = $V_{CC}$ or GND.
7. Add 3 mA per additional 1.0 MHz of operation over 1 MHz.
8. $C_L$ = 5 pF.

## Operating Conditions

| SYMBOL | PARAMETER | | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5 | 5.5 | 4.75 | 5 | 5.25 | V |
| $t_w$ | Width of clock | 20RS10A, 20RS8A, 20RS4A | 15 | 4 | | 15 | 4 | | ns |
| $t_{su}$ | Setup time for input to clock | | 20[1] | 11 | | 20[1] | 11 | | ns |
| $t_h$ | Hold time | | 0 | −10 | | 0 | −10 | | ns |
| $T_A$ | Operating free-air temperature | | −40 | 25 | 85 | 0 | 25 | 75 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{IL}$[2] | Low-level input voltage | | | | 0 | | 0.8 | V |
| $V_{IH}$[2] | High-level input voltage | | | | 2 | | $V_{CC}$ | V |
| $I_{IL}$ | Low-level input current | | $V_{CC}$ = MAX | $V_I$ = GND | | | −1 | μA |
| $I_{IH}$ | High-level input current | Pin 10[3] | $V_{CC}$ = MAX | $V_I$ = $V_{CC}$ | | 8 | 30 | μA |
| | | All other pins | | | | | 1 | μA |
| $V_{OL}$ | Low-level output voltage | | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.1 | 0.4 | V |
| | | | $V_{CC}$ = 5 V | $I_{OL}$ = 1 μA | | | 0.05 | |
| $V_{OH}$ | High-level output voltage | | $V_{CC}$ = MIN | $I_{OH}$ = −6 mA | 3.76[4] | 4.1 | | V |
| | | | $V_{CC}$ = 5 V | $I_{OH}$ = −1 μA | 4.95 | | | |
| $I_{OZL}$[5] | Off-state output current | | $V_{CC}$ = MAX | $V_O$ = GND | | 0 | −10 | μA |
| $I_{OZH}$[5] | | | | $V_O$ = $V_{CC}$ | | 0 | 10 | μA |
| $I_{CC}$ | Standby supply current[6] | | $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 0 | 100 | μA |
| | Operating supply current | | f = 1 MHz, $I_O$ = 0 mA, $V_I$ = GND or $V_{CC}$ | | | 2 | 5[7] | mA |

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS (See Test Load) | INDUSTRIAL MIN | INDUSTRIAL TYP | INDUSTRIAL MAX | COMMERCIAL MIN | COMMERCIAL TYP | COMMERCIAL MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output 20S10A, 20RS8A, 20RS4A | | $R_L$ = 1 KΩ, $C_L$ = 50 pF | | 13 | 25[1] | | 13 | 25[1] | ns |
| $t_{CLK}$ | Clock to output or feedback 20RS10A, 20RS8A, 20RS4A | | | | 8 | 15 | | 8 | 15 | ns |
| $t_{PZX}$ | Input to output enable | 20S10A, 20RS8A, 20RS4A | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] | Input to output disable | | | | 12 | 25 | | 12 | 25 | ns |
| $t_{PXZ}$[8] | Pin 13 (DIP) to output disable/enable | 20RS10A, 20RS8A, 20RS4A | | | 10 | 20 | | 10 | 20 | ns |
| $t_{PZX}$ | | | | | | | | | | |
| $f_{MAX}$ | Maximum frequency | | | 28.5 | 40 | | 28.5 | 40 | | MHz |

Notes:
1. For outputs with more than 12 inputs in a product term, $t_{SU}$ = 25 ns and $t_{PD}$ = 30 ns.
2. These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. Pin 10 DIP, pin 13 PLCC (PRELOAD pin). Applies to registered devices only.
4. JEDEC standard no. 7 for high-speed CMOS devices.
5. Applies to pins 14-23 for DIP (pins 17, 18, 20-27 for PLCC).
6. Disable output pins = $V_{CC}$ or GND.
7. Add 3 mA per additional 1.0 MHz of operation over 1 MHz.
8. $C_L$ = 5 pF.

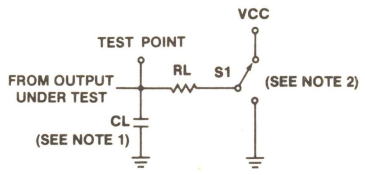
## Absolute Maximum Ratings

Supply voltage, $V_{CC}$ ..... -0.5 V to 7 V
DC input voltage, $V_I$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output voltage, $V_O$ ..... -0.5 V to $V_{CC}$ +0.5 V
DC output source/sink current per output pin, $I_O$ ..... ±35 mA
DC $V_{CC}$ or ground current, $I_{CC}$ or $I_{GND}$ ..... ±100 mA
Input diode current, $I_{IK}$:
- $V_I$<0 ..... -20 mA
- $V_I$>$V_{CC}$ ..... +20 mA

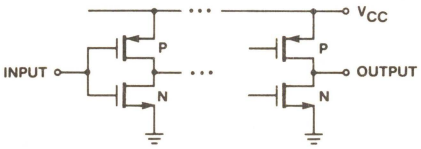
Output diode current, $I_{OK}$:
- $V_O$<0 ..... -20 mA
- $V_O$>$V_{CC}$ ..... +20 mA

Storage temperature ..... -65°C to +150°C

## Switching Test Load




## Schematic of Inputs and Outputs



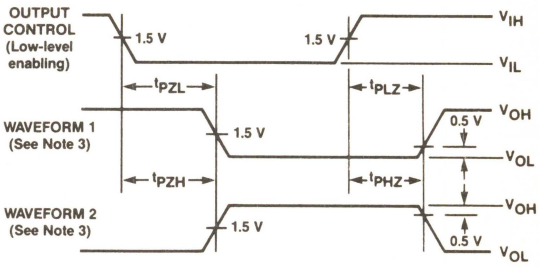

## Output Register PRELOAD†

The PRELOAD function allows the register to be loaded from data placed on the output pins. This feature aids functional testing of state sequencer designs by allowing direct setting of output states for improved test coverage. The procedure for PRELOAD using DIP pin numbers, is as follows:

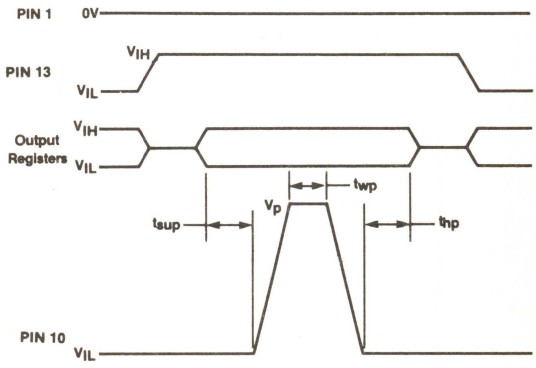
1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 13 to $V_{IH}$. Set pin 1 to 0 V.
3. Apply $V_{IL}$/$V_{IH}$ to all registered outputs.
4. Pulse pin 10 to $V_p$ (12 V), then back to 0 V.
5. Remove $V_{IL}$/$V_{IH}$ from all registered outputs.
6. Lower pin 13 to $V_{IL}$ to enable the output registers.
7. Verify for $V_{OL}$/$V_{OH}$ at all registered outputs.

† Note: Only applies to parts with output registers.
Typical $t_{sup}$ = 50 ns
$t_{wp}$ = 100 ns
$t_{hp}$ = 50 ns
$I_{IH}$ = 30μA (Pin 10)

## Enable/Disable Delay




Notes: 1. CL includes probe and jig capacitance.
2. When measuring $t_{PLZ}$ and $t_{PZL}$, S1 is tied to $V_{CC}$.
When measuring $t_{PHZ}$ and $t_{PZH}$, S1 is tied to ground.
$t_{PZX}$ is measured with $C_L$ = 50 pF. $t_{PXZ}$ is measured with $C_L$ = 5 pF.
When measuring propagation delay times of three-state outputs, S1 is open, i.e., not connected to $V_{CC}$ or ground.
3. Waveform 1 is for an output with internal conditions such that the output is LOW except when disabled by the output control.
Waveform 2 is for an output with internal conditions such that the output is HIGH except when disabled by the output control.



# ZHAL24A Evaluation #4

## Features/Benefits

- Demonstration pattern for ZHAL24A Series (ZHAL20X8A)
- 8-bit counter
- Three-state output
- Expandable in 8-bit increments
- Equivalent to 74ACT461

## Description

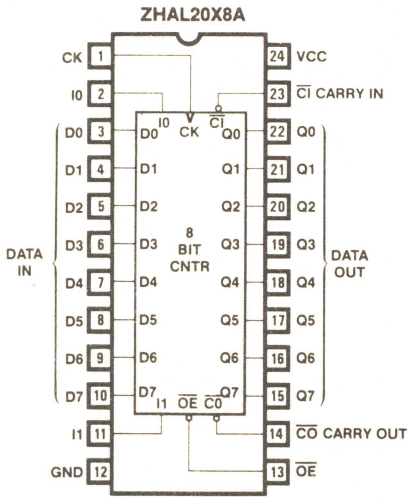
The ZHAL24A Evaluation #4 pattern is provided as an example of the features and characteristics of the ZHAL24A Series products. The design consists of an 8-bit synchronous counter with parallel load, clear, and hold capability. Two function select inputs (I0, I1) provide one of four operations which occur synchronously on the rising edge of the clock (CK).

The LOAD operation loads the inputs (D7-D0) into the output register (Q7-Q0). The CLEAR operation resets the output register to all LOWs. The HOLD operation holds the previous value regardless of clock transitions. The INCREMENT operation adds one to the output register when the carry-in input is TRUE ($\overline{CI}$ = LOW), otherwise the operation is a HOLD. The carry-out ($\overline{CO}$) is TRUE ($\overline{CO}$ = LOW) when the output register (Q7-Q0) is all HIGHs, otherwise FALSE ($\overline{CO}$ = HIGH).

The data output pins are enabled when $\overline{OE}$ is LOW, and disabled (HI-Z) when $\overline{OE}$ is HIGH.

Two or more 8-bit counters may be cascaded to provide larger counters. The operation codes were chosen such that when I1 is HIGH, I0 may be used to select between LOAD and INCREMENT as in a program counter (JUMP/INCREMENT).

## Logic Symbol




## Function Table

| $\overline{OE}$ | CK | I1 | I0 | $\overline{CI}$ | D7-D0 | Q7-Q0 | OPERATION |
|---|---|---|---|---|---|---|---|
| H | * | * | * | * | * | Z | HI-Z* |
| L | ↑ | L | L | X | X | L | CLEAR |
| L | ↑ | L | H | X | X | Q | HOLD |
| L | ↑ | H | L | X | D | D | LOAD |
| L | ↑ | H | H | H | X | Q | HOLD |
| L | ↑ | H | H | L | X | Q plus 1 | INCREMENT |

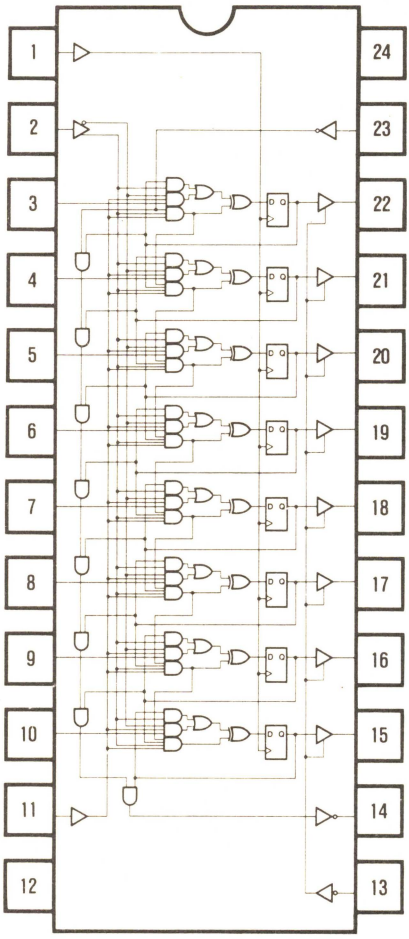
*When $\overline{OE}$ is HIGH, the three-state outputs are disabled to the high-impedance states; however, sequential operation of the counter is not affected.

H = HIGH voltage level
L = LOW voltage level
X = Don't care
Z = High impedance (off) state
↑ = LOW-to-HIGH clock transition

ZHAL™ is a trademark of Monolithic Memories.

## Logic Diagram

**ZHAL20X8A**



```
Title     ZHAL24A Evaluation 4 (74ACT461)
Pattern   P7023
Revision  B
Author    Birkner/Kazmi/Blasco
Company   Monolithic Memories, Inc.
Date      1986

CHIP ZHAL24A_Evaluation_4 PAL20X8

CK   I0 D0 D1 D2 D3 D4 D5 D6 D7  I1 GND
/OE /CO Q7 Q6 Q5 Q4 Q3 Q2 Q1 Q0 /CI VCC

EQUATIONS

/Q0 :=  /I1*/I0                                  ;CLEAR LSB
     +        I0*/Q0                             ;COUNT/HOLD
    :+:   I1*/I0*/D0                             ;LOAD D0 (LSB)
     +    I1* I0* CI                             ;COUNT

/Q1 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q1                             ;COUNT/HOLD
    :+:   I1*/I0*/D1                             ;LOAD D1
     +    I1* I0* CI*Q0                          ;COUNT

/Q2 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q2                             ;COUNT/HOLD
    :+:   I1*/I0*/D2                             ;LOAD D2
     +    I1* I0* CI*Q0*Q1                       ;COUNT

/Q3 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q3                             ;COUNT/HOLD
    :+:   I1*/I0*/D3                             ;LOAD D3
     +    I1* I0* CI*Q0*Q1*Q2                    ;COUNT

/Q4 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q4                             ;COUNT/HOLD
    :+:   I1*/I0*/D4                             ;LOAD D4
     +    I1* I0* CI*Q0*Q1*Q2*Q3                 ;COUNT

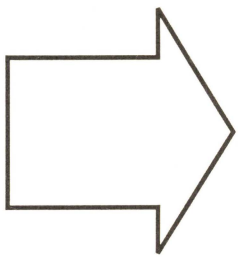
/Q5 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q5                             ;COUNT/HOLD
    :+:   I1*/I0*/D5                             ;LOAD D5
     +    I1* I0* CI*Q0*Q1*Q2*Q3*Q4              ;COUNT

/Q6 :=  /I1*/I0                                  ;CLEAR
     +        I0*/Q6                             ;COUNT/HOLD
    :+:   I1*/I0*/D6                             ;LOAD D6
     +    I1* I0* CI*Q0*Q1*Q2*Q3*Q4*Q5           ;COUNT

/Q7 :=  /I1*/I0                                  ;CLEAR MSB
     +        I0*/Q7                             ;COUNT/HOLD
    :+:   I1*/I0*/D7                             ;LOAD D7 (MSB)
     +    I1* I0* CI*Q0*Q1*Q2*Q3*Q4*Q5*Q6        ;COUNT

IF (VCC)  CO = CI*Q0*Q1*Q2*Q3*Q4*Q5*Q6*Q7        ;CARRY OUT
```

## Data Sheets



- TTL/CMOS PAL Devices
- TTL/CMOS AmPAL Devices
- PROSE/PLS Sequencers
- FPC/PEG Sequencers
- ECL PAL Devices
- HAL/ZHAL Devices
- Military PAL Devices
- Logic Cell Array
- Electrical Definitions


IMPORTANT
NOTES

# Military PAL Devices

Monolithic Memories' Military Programmable Array Logic (PAL) devices provide state machine and combinatorial logic solutions processed to military criteria. We offer the largest number of JAN 38510 and Standard Military Drawing PAL products in the industry.

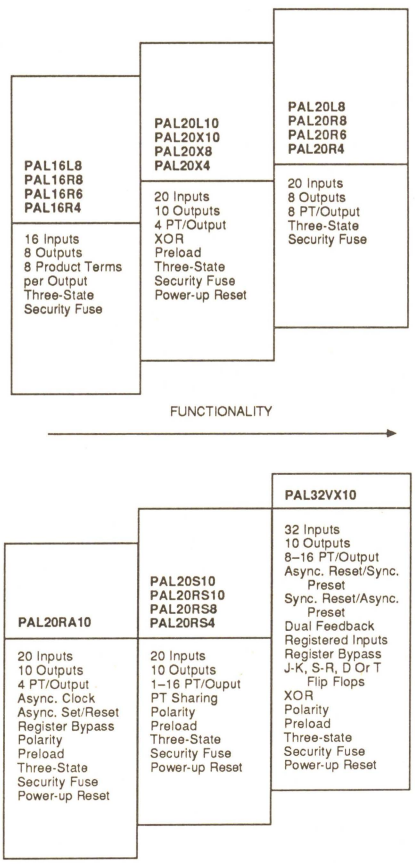
It has been stated that some level of radiation tolerance will be required in up to 50% of all military applications by 1990. The Military Products Division has started a Radiation Hardness Program. All of MMI's Bipolar processes passed neutron fluence testing up to $1x10^{13}$ neutrons per square centimeter. The junction isolated Bipolar processes also recovered in 50 to 70 microseconds from a 1 microsecond pulse of $2x10^{10}$RADs (Si) per second.

Applications for our configurable PAL architectures include counters, shift registers, accumulators, control sequence generators, decoders, multiplexers, adders, memory mapped I/O and much more. These designs go into radar systems, missile guidance, avionics, airport graphic terminals, parallel processors, military computer hardware, and product obsolescence solutions, just to name a few.

Military PAL devices go where you need:

| | | | |
|---|---|---|---|
| **High Speed** | **A Series** | **B Series** | **D Series** |
| | 30 nsec | 20 nsec | 15 nsec |

| | | |
|---|---|---|
| **Power Savings** | **Half Power** | **Quarter Power** |
| | 90 mA | 50 mA |




503 101

5

# Military PAL/PLD Device Menu

| DEVICE NAME | INPUTS | OUTPUTS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ in ns) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|
| PAL8L14A | 8 | 14 | 1 | 30 | 100 | 5-451 |
| PAL10H8 | 10 | 8 | 2 | 45 | 90 | 5-426 |
| PAL12H6 | 12 | 6 | 2,4 | 45 | 90 | 5-426 |
| PAL14H4 | 14 | 4 | 4 | 45 | 90 | 5-426 |
| PAL16H2 | 16 | 2 | 8 | 45 | 90 | 5-426 |
| PAL10L8 | 10 | 8 | 2 | 45 | 90 | 5-426 |
| PAL12L6 | 12 | 6 | 2,4 | 45 | 90 | 5-426 |
| PAL14L4 | 14 | 4 | 4 | 45 | 90 | 5-426 |
| PAL16L2 | 16 | 2 | 8 | 45 | 90 | 5-426 |
| PAL16C1 | 16 | 2 | 16 | 45 | 90 | 5-426 |
| PAL16L8D | 16* | 8 | 7 | 15 | 180 | 5-428 |
| PAL16L8B | | | | 20 | 180 | 5-428 |
| PAL16L8B-2 | | | | 30 | 90 | 5-430 |
| PAL16L8A | | | | 30 | 180 | 5-428 |
| PAL16L8B-4 | | | | 50 | 55 | 5-432 |
| PAL16L8A-2 | | | | 50 | 90 | 5-430 |
| PAL16L8A-4 | | | | 75 | 50 | 5-432 |
| AmPAL18P8B | 18* | 8 | 8 | 20 | 180 | 5-202 |
| AmPAL18P8AL | | | | 30 | 90 | 5-202 |
| AmPAL18P8A | | | | 30 | 180 | 5-202 |
| AmPAL18P8Q | | | | 40 | 55 | 5-202 |
| AmPAL18P8L | | | | 40 | 90 | 5-202 |
| PAL12L10 | 12 | 10 | 2 | 45 | 100 | 5-450 |
| PAL14L8 | 14 | 8 | 2,4 | 45 | 100 | 5-450 |
| PAL16L6 | 16 | 6 | 2,4 | 45 | 100 | 5-450 |
| PAL18L4 | 18 | 4 | 4,6 | 45 | 100 | 5-450 |
| PAL20L2 | 20 | 2 | 8 | 45 | 100 | 5-450 |
| PAL20C1 | 20 | 2 | 16 | 45 | 100 | 5-450 |
| PAL20L8B | 20* | 8 | 7 | 20 | 210 | 5-452 |
| PAL20L8A | | | | 30 | 210 | 5-452 |
| PAL20L8A-2 | | | | 50 | 105 | 5-454 |
| PAL20L10A | 20* | 10 | 3 | 35 | 165 | 5-456 |
| PAL20S10 | 20* | 10 | 0–16† | 40 | 240 | 5-458 |

* Includes feedback

† Product term steering

**Table 1. Simple Combinatorial PAL Devices**

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|---|
| PAL16R8D | 16* | 8 | 8 | 8 | 37 | 180 | 5-428 |
| PAL16R8B | | | | | 28.5 | 180 | 5-428 |
| PAL16R8B-2 | | | | | 20 | 90 | 5-430 |
| PAL16R8A | | | | | 20 | 180 | 5-428 |
| PAL16R8B-4 | | | | | 13.3 | 55 | 5-432 |
| PAL16R8A-2 | | | | | 13.3 | 90 | 5-430 |
| PAL16R8A-4 | | | | | 7.4 | 50 | 5-432 |
| PAL16R6D | 16* | 8 | 6 | 8 | 37 | 180 | 5-428 |
| PAL16R6B | | | | | 28.5 | 180 | 5-428 |
| PAL16R6B-2 | | | | | 20 | 90 | 5-430 |
| PAL16R6A | | | | | 20 | 180 | 5-428 |
| PAL16R6B-4 | | | | | 13.3 | 55 | 5-432 |
| PAL16R6A-2 | | | | | 13.3 | 90 | 5-430 |
| PAL16R6A-4 | | | | | 7.4 | 50 | 5-432 |
| PAL16R4D | 16* | 8 | 4 | 8 | 37 | 180 | 5-428 |
| PAL16R4B | | | | | 28.5 | 180 | 5-428 |
| PAL16R4B-2 | | | | | 20 | 90 | 5-430 |
| PAL16R4A | | | | | 20 | 180 | 5-428 |
| PAL16R4B-4 | | | | | 13.3 | 55 | 5-432 |
| PAL16R4A-2 | | | | | 13.3 | 90 | 5-430 |
| PAL16R4A-4 | | | | | 7.4 | 50 | 5-432 |
| PAL16X4 | 16* | 8 | 4 | 8† | 12 | 225 | 5-434 |
| PAL20R8B | 20* | 8 | 8 | 8 | 28.5 | 210 | 5-452 |
| PAL20R8A | | | | | 20 | 210 | 5-452 |
| PAL20R8A-2 | | | | | 13.3 | 105 | 5-454 |
| PAL20R6B | 20* | 8 | 6 | 8 | 28.5 | 210 | 5-452 |
| PAL20R6A | | | | | 20 | 210 | 5-452 |
| PAL20R6A-2 | | | | | 13.3 | 105 | 5-454 |
| PAL20R4B | 20* | 8 | 4 | 8 | 28.5 | 210 | 5-452 |
| PAL20R4A | | | | | 20 | 210 | 5-452 |
| PAL20R4A-2 | | | | | 13.3 | 105 | 5-454 |
| PAL20RS10 | 20* | 10 | 10 | 0–16†† | 16.7 | 240 | 5-458 |
| PAL20RS8 | 20* | 10 | 8 | 0–16†† | 16.7 | 240 | 5-458 |
| PAL20RS4 | 20* | 10 | 4 | 0–16†† | 16.7 | 240 | 5-458 |
| AmPAL22V10A | 22* | 10 | 0–10§ | 8–16§§ | 22 | 180 | 5-260 |
| AmPAL22V10 | | | | | 16.5 | 180 | 5-260 |

* Includes feedback
† Has an exclusive-OR gate
§ Flip-flops can be bypassed
†† Product term steering
§§ Has varied product term distribution

**Table 2. Simple Registered PAL Devices**

| DEVICE NAME | INPUTS | OUTPUTS | FLIP-FLOPS | FLIP-FLOP TYPES | PRODUCT TERMS/OUTPUT | SPEED ($f_{MAX}$ in MHz) | STAND BY $I_{CC}$ (mA) | DATASHEET PAGE NO. |
|---|---|---|---|---|---|---|---|---|
| PAL20X10A | 20* | 10 | 10 | D,T,JK,SR | 2/2† | 15.4 | 180 | 5-456 |
| PAL20X8A | 20* | 10 | 8 | D,T,JK,SR | 2/2† | 15.4 | 180 | 5-456 |
| PAL20X4A | 20* | 10 | 4 | D,T,JK,SR | 2/2† | 15.4 | 180 | 5-456 |
| AmPAL23S8-27 | 23* | 8 | 14§ | D,B◊ | 6–12§§ | 25 | 215 | 5-169 |
| AmPAL23S8-30 | | | | | | 22.5 | 215 | 5-169 |
| PAL32VX10A | 32* | 10 | 10§ | D,T,JK,SR, | 1/8–16† | 20 | 180 | 5-462 |
| PAL32VX10 | | | | B◊ | | 18 | 180 | 5-462 |

* Includes feedback
† Has an exclusive-OR gate
§ Some flip-flops can be bypassed
§§ Has varied product term distribution
◊ B=flip-flops are or can be buried

Table 3. State Machine PAL Devices

| DEVICE NAME | INPUTS | OUTPUTS | PRODUCT TERMS/OUTPUT | SPEED ($t_{PD}$ in ns) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|---|
| PAL20RA10 | 20* | 10 | 4 | 35** | 200 | 5-460 |

* Includes feedback
** With polarity fuse intact

Table 4. Asynchronous PAL Device

| DEVICE NAME | I/O PINS | CLBs | SPEED (INTERNAL TOGGLE $f_{MAX}$ in MHz) | STANDBY $I_{CC}$ (mA) | DATA SHEET PAGE NO. |
|---|---|---|---|---|---|
| M2064-50 | 58 | 64 | 50 | 5 | 5-518 |
| M2064-33 | | | 33 | 5 | 5-518 |
| M2064-20 | | | 20 | 5 | 5-518 |
| M2018-50 | | 100 | 50 | 5 | 5-518 |
| M2018-33 | 74 | | 33 | 5 | 5-518 |
| M2018-20 | | | 20 | 5 | 5-518 |

Table 5. LCA Devices

# Military 20-Pin PAL Devices

## Features

- Register and combinatorial outputs
- Variety of speed/power options
- Registers with feedback
- Programmable three-state outputs
- Security fuse prevents duplication of logic
- Through-hole or surface mount device packaging
- Neutron fluence (permanent damage): $1x10^{13}$ N/cm²
- Dose rate (transient upset) junction isolated Bipolar processes: $2x10^{10}$ RADs (Si) per sec recovered in 50 to 70 μs from a 1 μs pulse

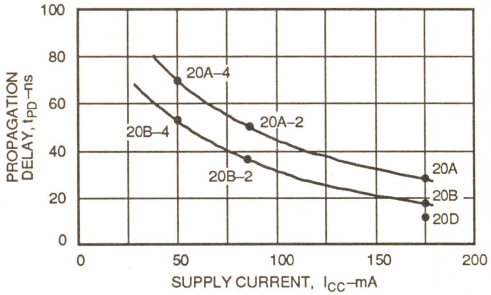
## Benefits

- Instant prototyping/zero NRE charge
- Low-cost programmable replacement for TTL logic
- Reduces inventory by reducing chip count
- Programmable on standard PROM/PAL device programmers
- Several software programs available to assist in creating bit pattern design

## Applications

- High speed graphic controllers
- High speed computers
- High frequency state machines
- High frequency counters
- Microprocessor clock generation and interface logic

**20-Pin PAL Device Speed vs Power**

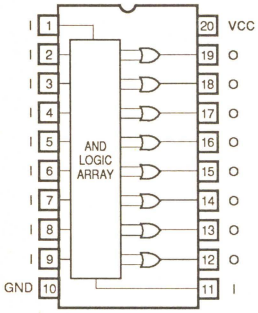



503 102

5

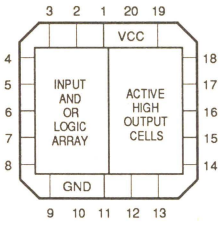
# Military 20-Pin PAL Device Pinouts

### 10H8/-2



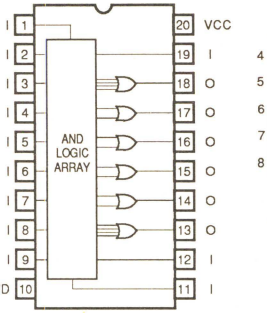

CERDIP CERPACK

### 10H8/-2



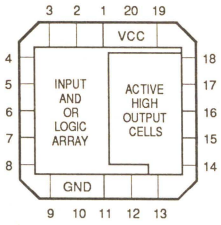

LEADLESS CHIP CARRIER

### 12H6/-2



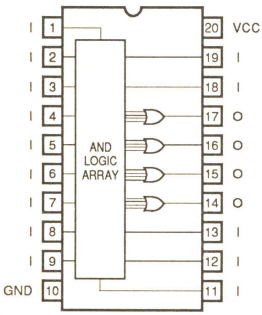

CERDIP CERPACK

### 12H6/-2



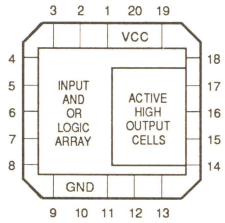

LEADLESS CHIP CARRIER

### 14H4/-2



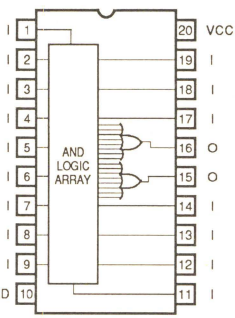

CERDIP CERPACK

### 14H4/-2



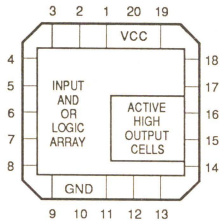

LEADLESS CHIP CARRIER

### 16H2/-2



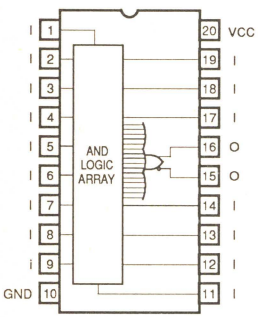

CERDIP CERPACK

### 16H2/-2



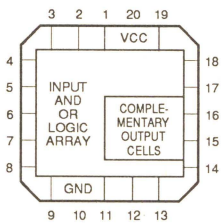

LEADLESS CHIP CARRIER

### 16C1/-2



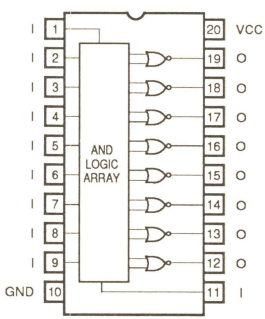

CERDIP CERPACK

### 16C1/-2



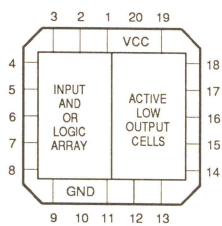

LEADLESS CHIP CARRIER

### 10L8/-2




CERDIP CERPACK

### 10L8/-2

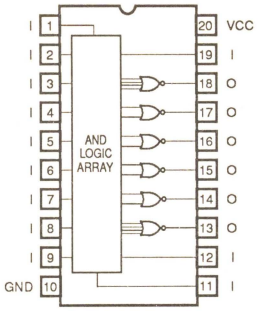



LEADLESS CHIP CARRIER

503 103

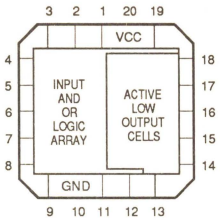
# Military 20-Pin PAL Device Pinouts

### 12L6/-2



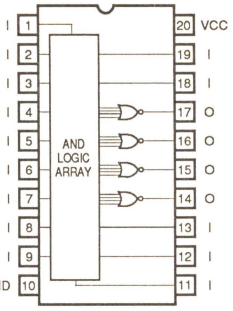

CERDIP CERPACK

### 12L6/-2



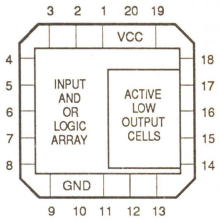

LEADLESS CHIP CARRIER

### 14L4/-2



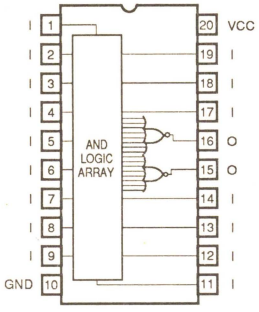

CERDIP CERPACK

### 14L4/-2



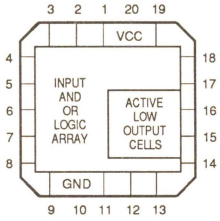

LEADLESS CHIP CARRIER

### 16L2/-2



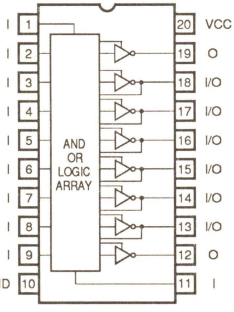

CERDIP CERPACK

### 16L2/-2



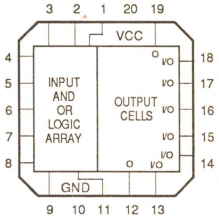

LEADLESS CHIP CARRIER

### 16L8D/B/B-2/B-4 A/A-2/A-4



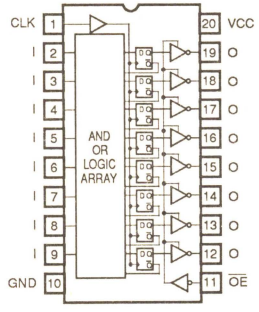

CERDIP CERPACK

### 16L8D/B/B-2/B-4 A/A-2/A-4



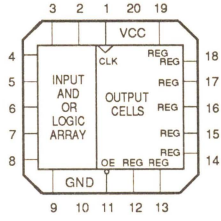

LEADLESS CHIP CARRIER

### 16R8D/B/B-2/B-4 A/A-2/A-4



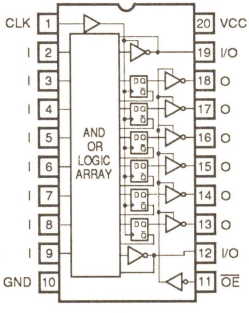

CERDIP CERPACK

### 16R8D/B/B-2/B-4 A/A-2/A-4



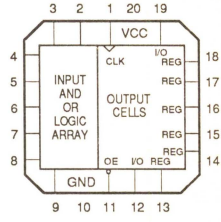

LEADLESS CHIP CARRIER

### 16R6D/B/B-2/B-4 A/A-2/A-4




CERDIP CERPACK

### 16/R6D/B/B-2/B-4 A/A-2/A-4

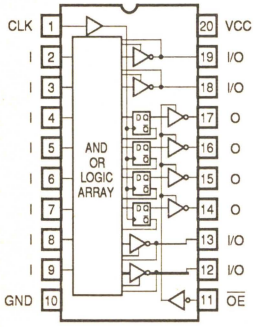



LEADLESS CHIP CARRIER

503 115

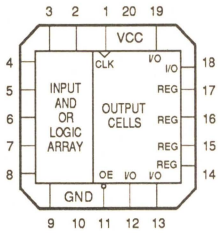
# Military 20-Pin PAL Device Pinouts

**16R4D/B/B-2/B-4**
**A/A-2/A-4**



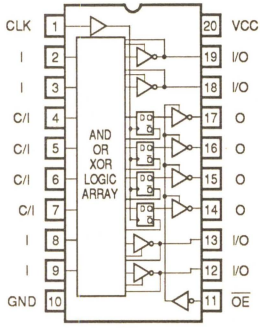

CERDIP CERPACK

**16R4D/B/B-2/B-4**
**A/A-2/A-4**



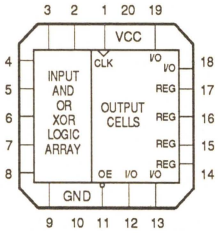

LEADLESS CHIP CARRIER

**16X4**




CERDIP CERPACK

**16X4**




LEADLESS CHIP CARRIER

503 127

# Military 20-Pin PAL Devices

## Absolute Maximum Ratings

| | Operating |
|---|---|
| Supply voltage, VCC | –0.5 V to 7 V |
| Input voltage range | –1.5 V to 5.5 V |
| Off-state output voltage | 5.5 V |
| Storage temperature | –65°C to +150°C |
| Maximum junction temperature ($T_j$) | 175°C |
| Lead temperature (soldering, 10 sec max) | 300°C |
| Maximum current density $5x10^{-5}$ A/cm² per Mil-M-38510 | < $5x10^{-5}$ A/cm² |
| Maximum $\theta_{jc}$ = 28°C/W for cerdips per Mil-M-38510 | < 28°C/W |
| Maximum $\theta_{jc}$ = 22°C/W for flatpacks per Mil-M-38510 | < 22°C/W |
| Maximum $\theta_{jc}$ = 20°C/W for leadless chip carrier per Mil-M-38510 | < 20°C/W |

# Military Standard 20-Pin PAL Series

## PAL10H8, 12H6, 14H4, 16H2, 16C1, 10L8, 12L6, 14L4, 16L2

Can be purchased to standard military drawings 81035, latest revision in effect.

# Military 20-Pin Half-Power PAL Series

## PAL10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2, 16L2-2

### Operating Conditions

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | –55 | | °C |
| $T_C$ | Operating case temperature | | 125 | °C |
| $V_{IL}$* | Low-level input voltage | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | 10H8, 12H6, 14H4, 16H2,16C1, 10L8, 12L6, 14L4, 16L2 | | 0.5 | V |
| | | $I_{OL}$ = 4 mA<br>$I_{OH}$ = –2 mA | 10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2, 16L2-2 | | | |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | 10H8, 12H6, 14H4, 16H2,16C1, 10L8, 12L6, 14L4, 16L2 | 2.4 | | V |
| | | $I_{OH}$ = –1 mA<br>$I_{OH}$ = –2 mA | 10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2, 16L2-2 | | | |
| $I_{OS}$* | Output short-circuit current | $V_{CC}$ = 5V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | 10H8, 12H6, 14H4, 16H2, 16C1, 10L8, 12L6, 14L4, 16L2 | | 90 | mA |
| | | | 10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2, 16L2-2 | | 45 | |

* Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Standard 20-Pin PAL Series

## PAL10H8, 12H6, 14H4, 16H2, 16C1, 10L8, 12L6, 14L4, 16L2

# Military 20-Pin Half-Power PAL Series

## PAL10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2, 16L2-2

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | 10H8, 12H6, 14H4, 16H2, 16C1, 10L8, 12L6, 14L4, 16L2 | $R_1$ = 560 Ω<br>$R_2^1$ = 1.1 KΩ | | 45 | ns |
| | | 10H8-2, 12H6-2, 14H4-2, 16H2-2, 16C1-2, 10L8-2, 12L6-2, 14L4-2,16L2-2 | $R_1$ = 1.12 KΩ<br>$R_2^1$ = 2.2 KΩ | | 80 | ns |

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10, and 11.

5

# Military Ultra High Speed 20-Pin PAL Series

## PAL16L8D, 16R8D, 16R6D, 16R4D

Can be purchased to standard military drawing 5962-85155, latest revision in effect.

# Military Very High Speed 20-Pin PAL Series

## PAL16L8B, 16R8B, 16R6B, 16R4B

Can be purchased to standard military drawing 5962-85155, latest revision in effect.

# Military High Speed 20-Pin PAL Series

## PAL16L8A, 16R8A, 16R6A, 16R4A

Can be purchased to standard military drawing 81036, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | 20 D | | 20 B | | 20 A | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | 4.5 | 5.5 | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | –55 | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | | 125 | | 125 | °C |
| $t_w$† | Width of clock (except 16L8) | Low | 12 | | 12 | | 20 | | ns |
| | | High | 8 | | 12 | | 20 | | |
| $t_{su}$† | Set up time from input or feedback to clock (except 16L8) | | 15 | | 20 | | 30 | | ns |
| $t_h$† | Hold time | | 0 | | 0 | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | | ≤0.8 | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | ≥2.0 | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.
* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.
† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 180 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.
** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Ultra High Speed 20-Pin PAL Series

**PAL16L8D, 16R8D, 16R6D, 16R4D**

# Military Very High Speed 20-Pin PAL Series

**PAL16L8B, 16R8B, 16R6B, 16R4B**

# Military High Speed 20-Pin PAL Series

**PAL16L8A, 16R8A, 16R6A, 16R4A**

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | 20 D | | 20 B | | 20 A | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{PD}$ | Input or feedback to output (except 16R8) | $R_1$ = 390 Ω, $R_2$ = 750 Ω | | 15 | | 20 | | 30 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 16L8) | | | 12 | | 15 | | 20 | ns |
| $t_{PZX}$ | Pin 11 to output enable (except 16L8) | | | 12 | | 20 | | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable (except 16L8) | | | 10 | | 20 | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 16R8) | | | 17 | | 25 | | 30 | ns |
| $t_{PXZ}$ | Input to output disable (except 16R8) | | | 13 | | 20 | | 30 | ns |
| $f$MAX* | State machine maximum operating frequency (except 16L8) | | 37 | | 28.5 | | 20 | | MHz |
| | Data path register maximum operating frequency (except 16L8) | | 50 | | 41.6 | | 25 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1,2,3,7,8,9,10 and 11.

# Military Half-Power 20B-Pin Series

## PAL16L8B-2, 16R8B-2, 16R6B-2, 16R4B-2

Can be purchased to standard military drawing 5962-85155, latest revision in effect.

# Military Half-Power 20A-Pin Series

## PAL16L8A-2, 16R8A-2, 16R6A-2, 16R4A-2

Can be purchased to standard military drawing 81036, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | 20 B–2 | | 20 A–2 | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | –55 | 125 | °C |
| $T_C$ | Operating case temperature | | | 125 | | | °C |
| $t_w$† | Width of clock (except 16L8) | Low | 20 | | 25 | | ns |
| | | High | 20 | | 25 | | |
| $t_{su}$† | Set up time from input or feedback to clock (except 16L8) | | 30 | | 50 | | ns |
| $t_h$† | Hold time | | 0 | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | 16L8B-2, 16R8B-2, 16R6B-2, 16R4B-2 | | –250 | mA |
| | | $V_O$ = 0.5 V | 16L8A-2, 16R8A-2, 16R6A-2, 16R4A-2 | –30 | –130 | |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 90 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Half-Power 20B-Pin Series

## PAL16L8B-2, 16R8B-2, 16R6B-2, 16R4B-2

Can be purchased to standard military drawing 5962-85155, latest revision in effect.

# Military Half-Power 20A-Pin Series

## PAL16L8A-2, 16R8A-2, 16R6A-2, 16R4A-2

Can be purchased to standard military drawing 81036, latest revision in effect.

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | 20 B-2 | | 20 A-2 | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $t_{PD}$ | Input or feedback to output (except 16L8) | $R_1$ = 390 Ω<br>$R_2$ = 750 Ω | | 30 | | 50 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 16L8) | | | 20 | | 25 | ns |
| $t_{PZX}$ | Pin 11 to output enable (except 16L8) | | | 25 | | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable (except 16L8) | | | 25 | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 16R8) | | | 30 | | 45 | ns |
| $t_{PXZ}$ | Input to output disable (except 16R8) | | | 30 | | 45 | ns |
| *f*MAX** | State machine maximum operating frequency (except 16L8) | | 20 | | 13.3 | | MHz |
| | Data path register maximum operating frequency (except 16L8) | | 25 | | 20 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1,2,3,7,8,9,10 and 11.

5

# Military Quarter-Power 20B-Pin Series

## PAL16L8B-4, 16R8B-4, 16R6B-4, 16R4B-4

Can be purchased to standard military drawing 5962-88515 latest revision in effect.

# Military Quarter-Power 20A-Pin Series

## PAL16L8A-4, 16R8A-4, 16R6A-4, 16R4A-4

Can be purchased to standard military drawing 85065 latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | 20 B-4 | | 20 A-4 | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | –55 | 125 | °C |
| $T_C$ | Operating case temperature | | | 125 | | | °C |
| $t_w$† | Width of clock (except 16L8) | Low | 25 | | 40 | | ns |
| | | High | 25 | | 40 | | |
| $t_{su}$ | Set up time from input or feedback to clock (except 16L8) | | 50 | | 90 | | ns |
| $t_h$† | Hold time | | 0 | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 4 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –1 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | 16L8B-4, 16R8B-4, 16R6 B-4, 16R4B-4 | | –250 | mA |
| | | $V_O$ = 0.5 V | 16L8A-4, 16R8A-4, 16R6A-4, 16R4A-4 | –30 | –130 | |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | 16L8A-4, 16R8A-4, 16R6A-4, 16R4A-4 | | 50 | mA |
| | | | 16L8B-4, 16R8B-4, 16R6B-4, 16R4B-4 | | 55 | |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Quarter-Power 20B-Pin Series

## PAL16L8B-4, 16R8B-4, 16R6B-4, 16R4B-4

# Military Quarter-Power 20A-Pin Series

## PAL16L8A-4, 16R8A-4, 16R6A-4, 16R4A-4

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | 20 B-4 | | 20 A-4 | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $t_{PD}$ | Input or feedback to output (except 16R8) | $R_1$ = 800 Ω<br>$R_2$ = 1.56 KΩ | | 50 | | 75 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 16L8) | | | 25 | | 45 | ns |
| $t_{PXZ}$ | Pin 11 to output enable(except 16L8) | | | 25 | | 40 | ns |
| $t_{PXZ}$ | Pin 11 to output disable (except 16L8) | | | 25 | | 40 | ns |
| $t_{PZX}$ | Input to output enable (except 16R8) | | | 45 | | 65 | ns |
| $t_{PXZ}$ | Input to output disable (except 16R8) | | | 45 | | 65 | ns |
| *f*MAX* | State machine maximum operating frequency (except 16L8) | | 13.3 | | 7.4 | | MHz |
| | Data path register maximum operating frequency (except 16L8) | | 20 | | 12.5 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1,2,3,7,8,9,10 and 11.

5

# Military Arithmetic 20-Pin PAL Series

## PAL16X4

### Operating Conditions

| SYMBOL | PARAMETER | | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | °C |
| $t_w$† | Width of clock | Low | 25 | | ns |
| | | High | 25 | | |
| $t_{su}$† | Set up time from input or feedback to clock | | 55 | | ns |
| $t_h$† | Hold time | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | 16X 4 | | 225 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Arithmetic 20-Pin PAL Devices

## PAL16X4

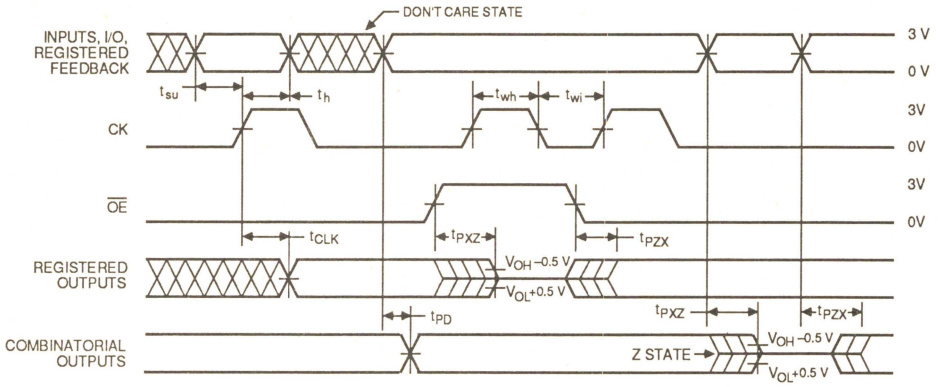
### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | | 45 | ns |
| $t_{CLK}$ | Clock to output or feedback | | | 25 | ns |
| $t_{PZX}$ | Pin 11 to output enable | $R_1 = 200\ \Omega$ | | 25 | ns |
| $t_{PXZ}$ | Pin 11 to output disable | $R_2 = 390\ \Omega$ | | 25 | ns |
| $t_{PZX}$ | Input to output enable | | | 45 | ns |
| $t_{PXZ}$ | Input to output disable | | | 45 | ns |
| $f$MAX | Maximum frequency | | 12.5 | | MHz |

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1,2,3,7,8,9,10 and 11.

# Military 20-Pin PAL Devices

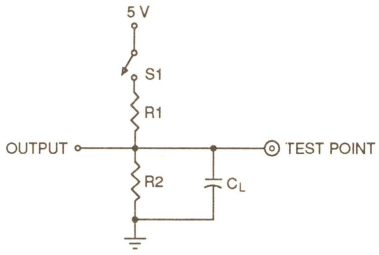
## Switching Waveforms




503 133

Notes:
1. $t_{PD}$ is tested with switch $S_1$ closed. $C_L$ = 50 pF and measured at 1.5 V output level.
2. $t_{PZX}$ is measured at the 1.5 V level with $C_L$ = 50 pF. $S_1$ is open for high impedance to "1" test, and closed for high impedance to "o" test.
3. $t_{PXZ}$ is tested with $C_L$ = 5 pF. $S_1$ is open for "1" to high impedance test, measured $V_{OH}$ –0.5 V output level. $S_1$ is closed for "0" to high impedance test measured to $V_{OL}$ +0.5 V output level.
4. Equivalent test loads may be used on automatic test equipment.

## Test Load




503 194

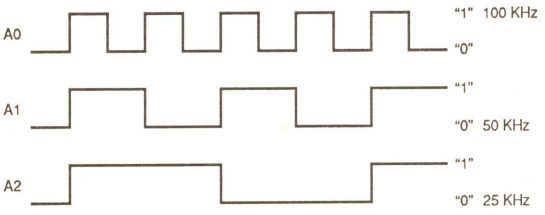
# Military 20-Pin PAL Devices

## Life Test/Burn-In Circuits

Complies with Mil-Std-883, Method 1005/1015, Condition D.

## Circuit Configurations

## Waveforms

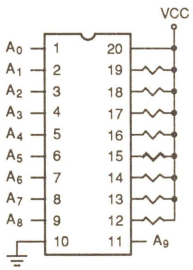



A3 to Ax Follow in order

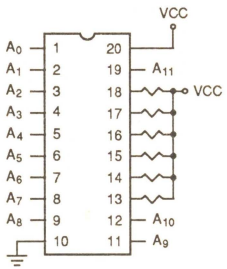
503 134

1. All Burn-In will be accomplished at 125° C +5/–0°C
2. $V_{CC}$ = 5.25 Volts ± 0.25 V
3. All clocks (A0 to Ax) are square wave signals, 50±15% Duty Cycle, with:
   a. "0" = –0.5 V to +0.7 V
   b. "1" = +2.4 V to $V_{CC}$
   c. Rise Time (+0.7 V to +2.4 V) < 1 µsec
   d. Fall Time (+2.4 V to +0.7 V) < 1 µsec
4. Resistor Value
   330 Ω or 470 Ω ±5%
5. All Board Components to be compatible with 150°C Ambient (Min).

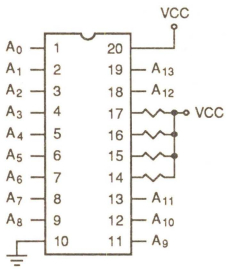
**PAL10H8/H8-2**
**PAL10L8/L8-2**




**PAL12H6/H6-2**
**PAL12L6/L6-2**




**PAL14H4/H4-2**
**PAL14L4/L4-2**




503 135

5

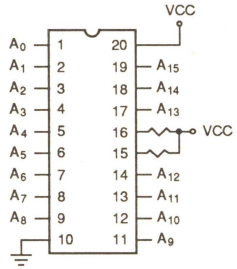
# Military 20-Pin PAL Devices

## Life Test/Burn-In Circuits

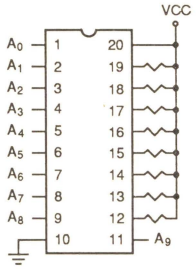
Complies with Mil-Std-883, Method 1005/1015, Condition D.

## Circuit Configurations

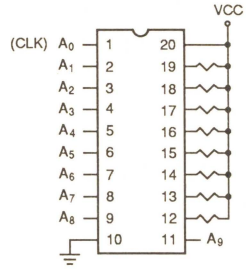
**PAL16H2/H2-2**
**PAL16L2/L2-2**
**PAL16C1/C1-2**



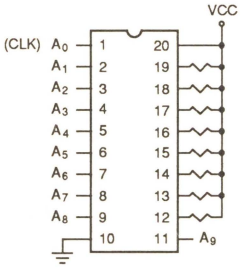

**PAL16L8A/B/D**
**PAL16L8A-2/B-2**
**PAL16L8A-4/B-4**



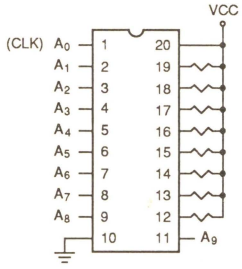

**PAL16R8A/B/D**
**PAL16R8A-2/B-2**
**PAL16R8A-4/B-4**



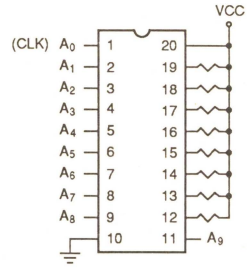

**PAL16R6A/B/D**
**PAL16R6A-2/B-2**
**PAL16R6A-4/B-4**




**PAL16R4A/B/D**
**PAL16R4A-2/B-2**
**PAL16R4A-4/B-4**




**PAL16A4**
**PAL16X4**



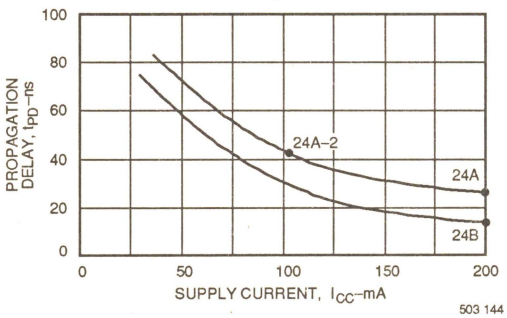

503 138

# Military 24-Pin PAL Devices

## Features

- Registers with feedback
- Programmable three-state outputs
- Security fuse prevents duplication of logic
- Variety of speed/power options available in same architecture
- Register preload to aid in device testing
- Power-up reset to logical high
- Programmable output polarity
- Product term sharing
- Programmable register or combinatorial outputs
- Dual feedback allows buried state registers or input registers (PAL32VX10)
- Programmable flip-flops allow J-K, S-R, T or D types (PAL32VX10)
- Asynchronous preset/synchronous reset, synchronous preset/asynchronous reset (PAL32VX10)
- Through-hole or surface mount device packaging
- Neutron fluence (permanent damage): $1x10^{13}$ N/cm²
- Dose rate (transient upset) junction isolated Bipolar processes: $2x10^{10}$ RADs (Si) per sec recovered in 50 to 70 μs from a 1μs pulse

**24-Pin PAL Device Speed vs. Power**




503 144

## Benefits

- Instant prototyping/zero NRE charge
- Low-cost programmable replacement for TTL logic
- Reduces inventory by reducing chip count
- Programmed on standard PROM/PAL device programmers
- Several software programs available to assist in creating bit pattern designs

## Applications

- High speed graphic controllers
- High speed computers
- High frequency state machines
- High frequency counters
- Microprocessor clock generation and interface logic
- DMA controllers
- Asynchronous bus interface
- CRT controllers
- Peripheral/handshaking interface
- Interrupt controllers
- Memory mapped I/O (PAL8L14A)
- Microprocessor decoder (PAL8L14A)

5

## Register Preload

Register preload is an aid to functional testing, which is usually performed after the device is programmed but before it is installed on the circuit board. Using register preload, the register of a device can be "preloaded" to any desired state value. This is particularly useful in applications where the output is fed back into the array as an input, since it may take many state transitions to reach a desired state in the output register. Register preload also allows the user to set the device to an "illegal" state which cannot be reached through normal state transitions, in order to test for proper recovery.

## Power-Up Reset

Another added testability feature found on these Series is power-up reset. Power-up reset makes system initialization simple; registers are reset to logic 0 at power-up, thus all outputs are set to logic 1.

The table below is a brief summary of our current devices that do have register preload and/or power-up reset.

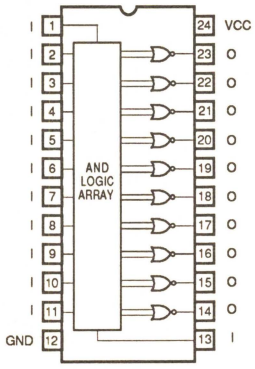
## Devices with Register Preload and Power-Up Reset

| DEVICE FAMILY | REGISTER PRELOAD | POWER-UP RESET |
|---|---|---|
| Exclusive OR 24XA | YES | YES |
| Shared Product terms 24RS | YES | YES |
| Asynchronous 24RA | YES | YES |
| Varied Product terms 24VX* | YES | YES |

* The PAL32VX10/10A has power-up preset; registers are set to logical 1 on power up.

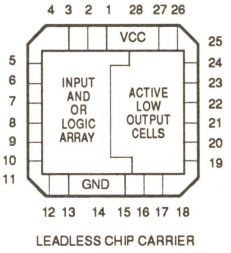
# Military 24-Pin PAL Device Pinouts

### 12L10



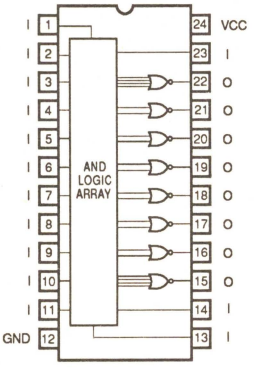

CERDIP CERPACK

### 12L10



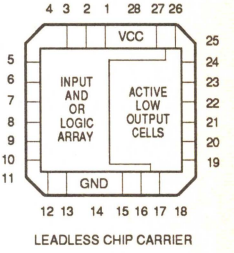

LEADLESS CHIP CARRIER

### 14L8



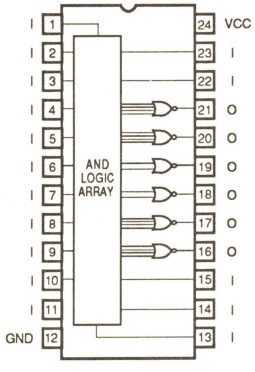

CERDIP CERPACK

### 14L8



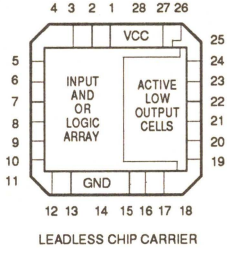

LEADLESS CHIP CARRIER

### 16L6



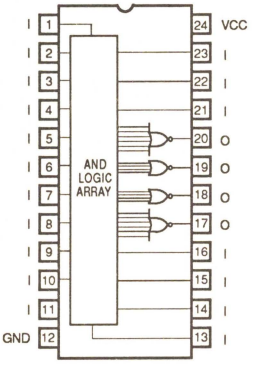

CERDIP CERPACK

### 16L6



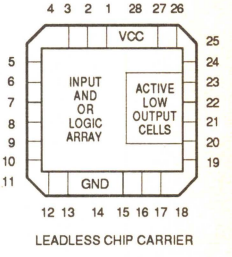

LEADLESS CHIP CARRIER

### 18L4



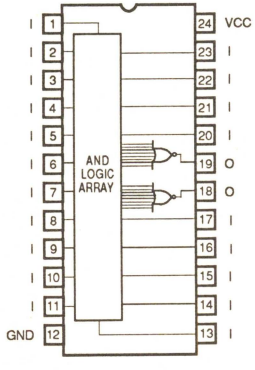

CERDIP CERPACK

### 18L4



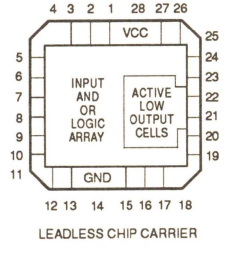

LEADLESS CHIP CARRIER

### 20L2



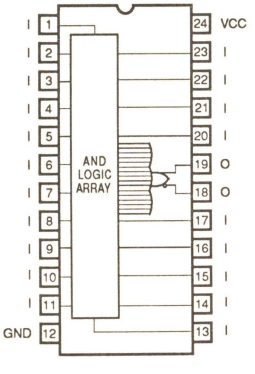

CERDIP CERPACK

### 20L2



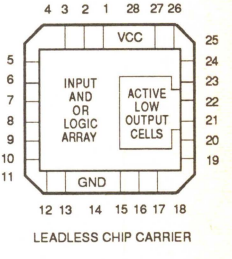

LEADLESS CHIP CARRIER

### 20C1




CERDIP CERPACK

### 20C1

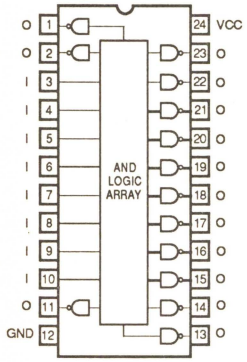



LEADLESS CHIP CARRIER

503 145

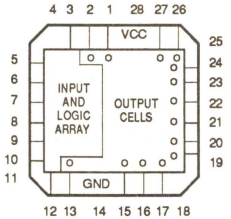
# Military 24-Pin PAL Device Pinouts

### 8L14A



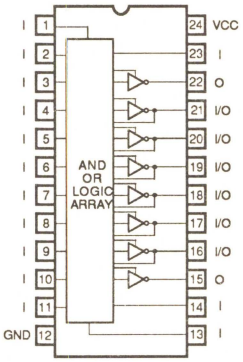

CERDIP CERPACK

### 8L14A



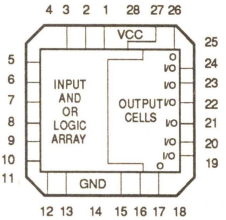

LEADLESS CHIP CARRIER

### 20L8B/A/A-2



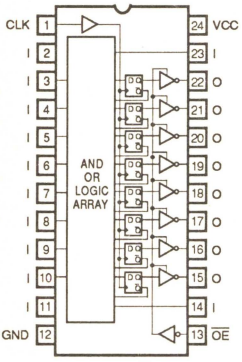

CERDIP CERPACK

### 20L8B/A/A-2



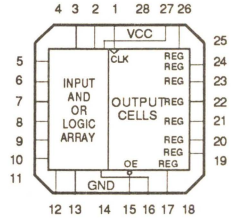

LEADLESS CHIP CARRIER

### 20R8B/A/A-2



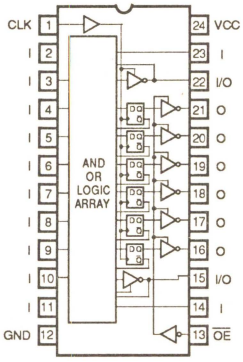

CERDIP CERPACK

### 20R8B/A/A-2



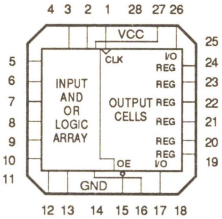

LEADLESS CHIP CARRIER

### 20R6B/A/A-2



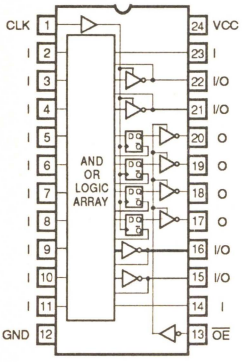

CERDIP CERPACK

### 20R6B/A/A-2



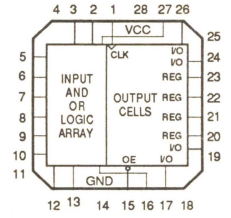

LEADLESS CHIP CARRIER

### 20R4B/A/A-2



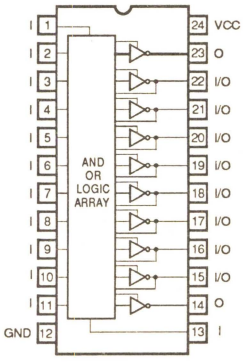

CERDIP CERPACK

### 20R4B/A/A-2



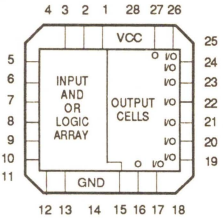

LEADLESS CHIP CARRIER

### 20L10A




CERDIP CERPACK

### 20L10A

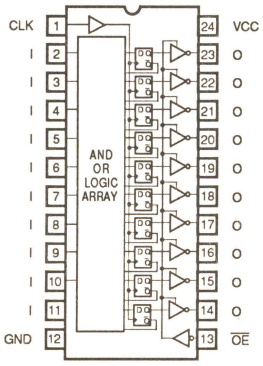



LEADLESS CHIP CARRIER

503 157

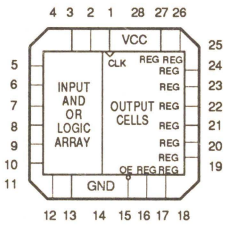
# Military 24-Pin PAL Device Pinouts

### 20X10A



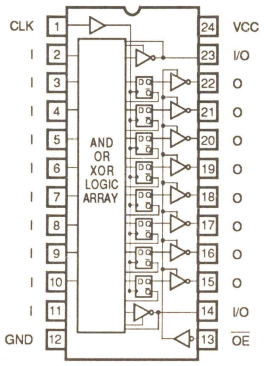

CERDIP CERPACK

### 20X10A



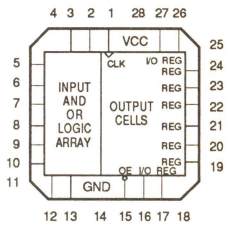

LEADLESS CHIP CARRIER

### 20X8A



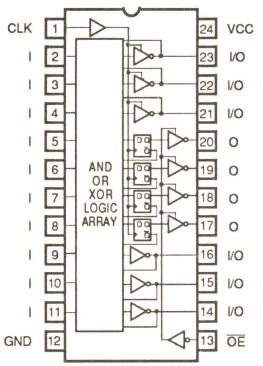

CERDIP CERPACK

### 20X8A



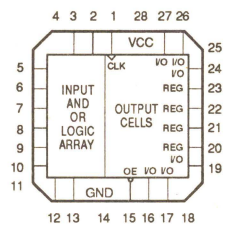

LEADLESS CHIP CARRIER

### 20X4A



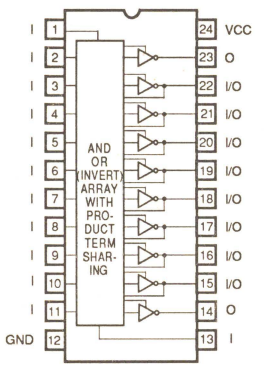

CERDIP CERPACK

### 20X4A



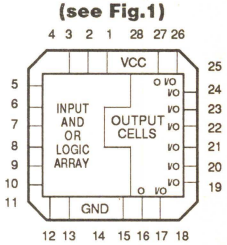

LEADLESS CHIP CARRIER

### 20S10 (see Fig. 1)



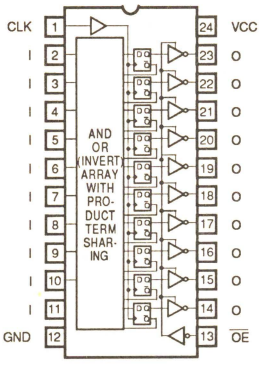

CERDIP CERPACK

### 20S10 (see Fig.1)



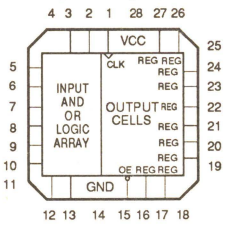

LEADLESS CHIP CARRIER

### 20RS10 (see Fig. 1)



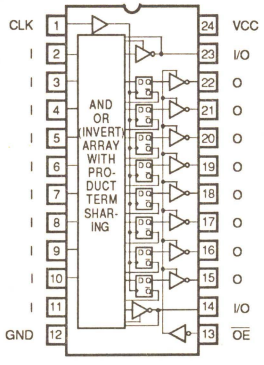

CERDIP CERPACK

### 20RS10 (see Fig.1)



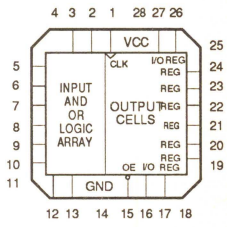

LEADLESS CHIP CARRIER

### 20RS8 (see Fig 1)




CERDIP CERPACK

### 20RS8 (see Fig.1)



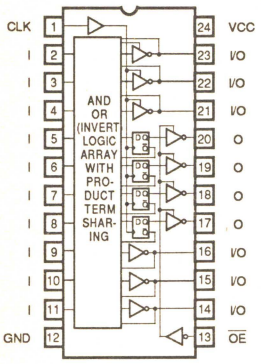

LEADLESS CHIP CARRIER

503 169

5

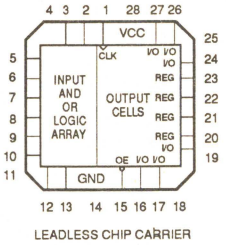
# Military 24-Pin PAL Device Pinouts

**20RS4**
**(see Fig.1)**



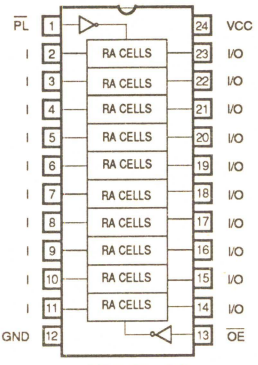

CERDIP CERPACK

**20RS4**
**(see Fig. 1)**



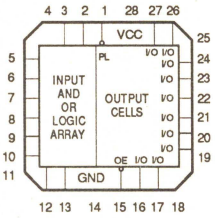

LEADLESS CHIP CARRIER

**20RA10**
**(see Fig.2)**



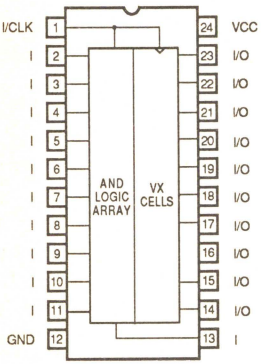

CERDIP CERPACK

**20RA10**
**(see Fig. 2)**



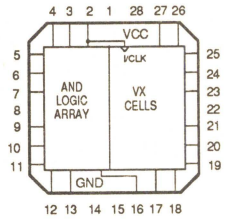

LEADLESS CHIP CARRIER

**32VX10/A**
**(see Fig. 3)**




CERDIP CERPACK

**32VX10/A**
**(see Fig 3)**




LEADLESS CHIP CARRIER

503 181

# Military PAL20S10, 20RS10, 20RS8, 20RS4 Series

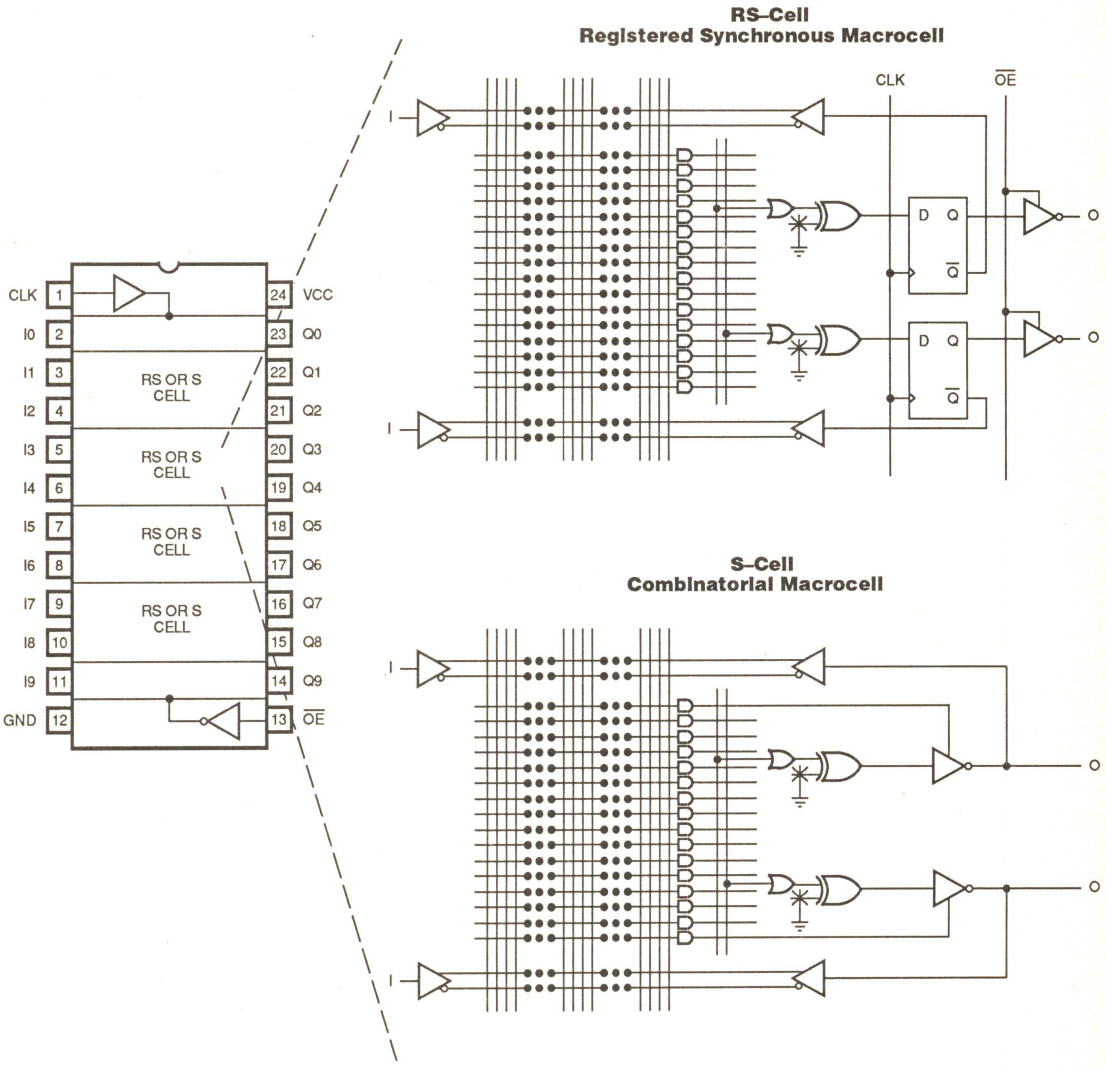



Figure 1.

503 187

5

# Military PAL20RA10 Device

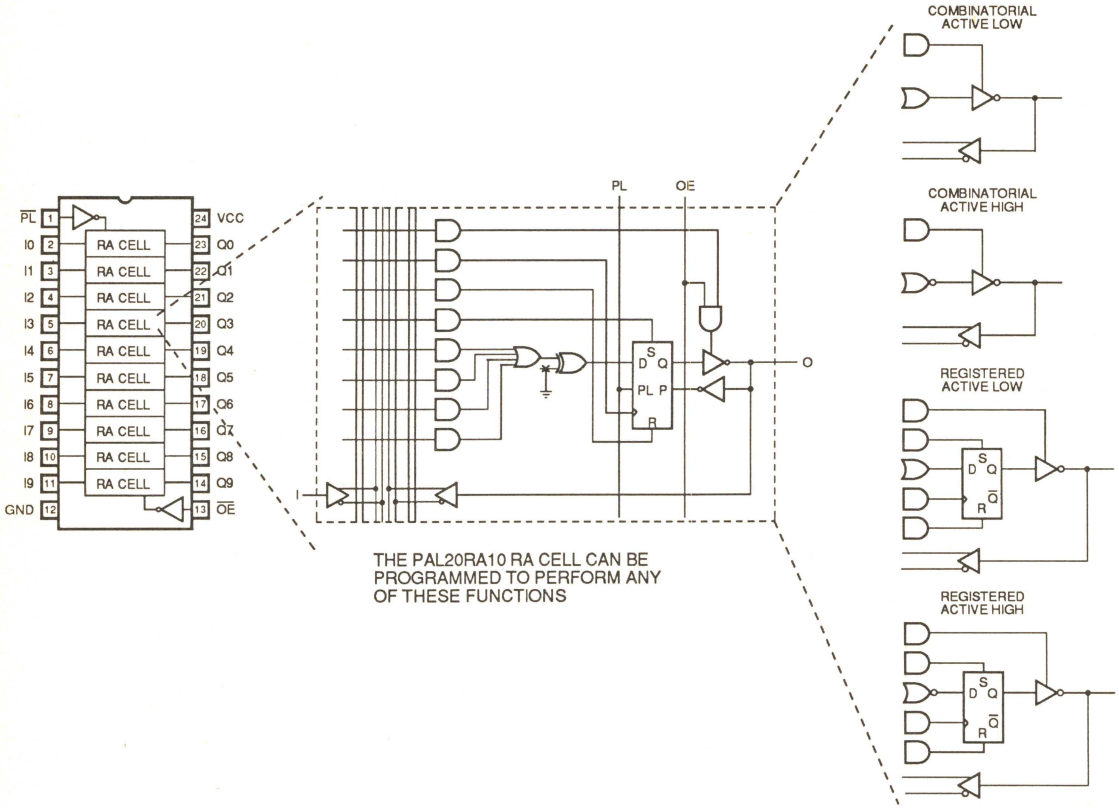



Figure 2.

# Military PAL32VX10 Device

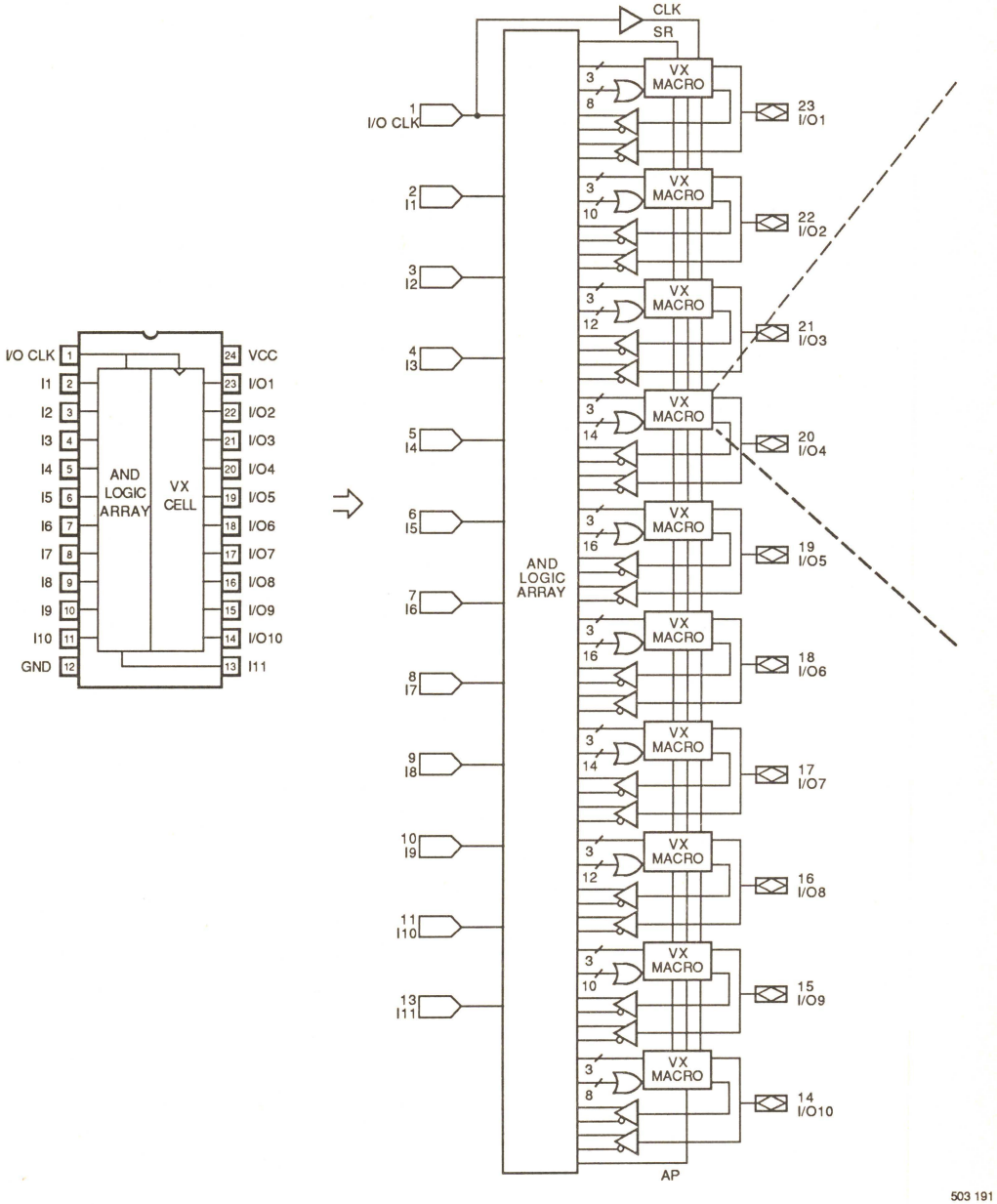
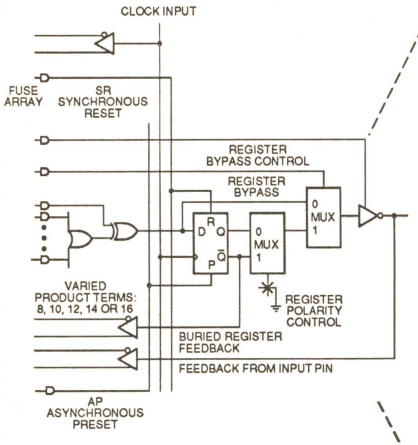
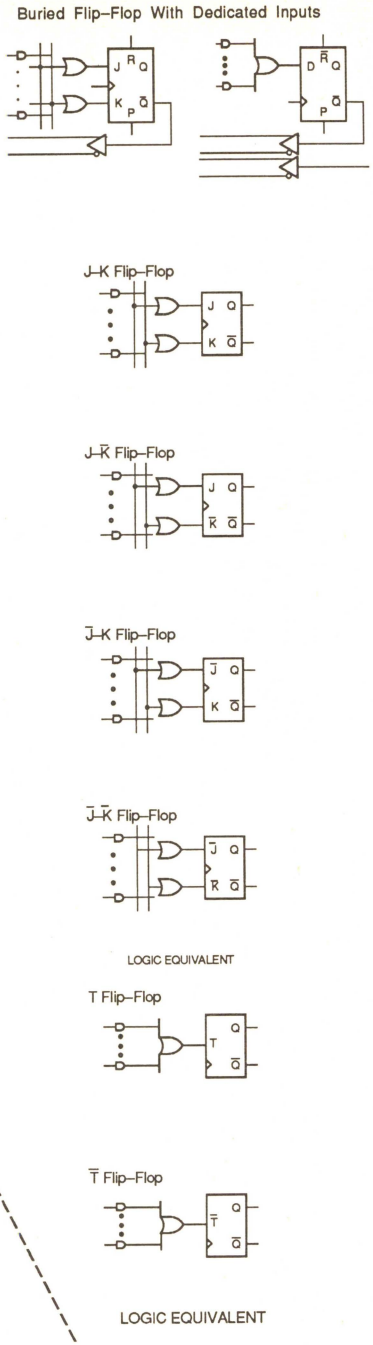



Figure 3.




PAL32VX10 MACROCELL

THE PAL32VX10 MACROCELL CAN BE PROGRAMMED TO ANY OF THESE FUNCTIONS.




503 192

Figure 3. (Cont'd.)

# Military 24-Pin PAL Devices

## Absolute Maximum Ratings

| | Operating |
|---|---|
| Supply voltage $V_{CC}$ | –0.5 V to 7 V |
| Input voltage | –1.5 V to 5.5 V |
| Off-state output voltage | 5.5 V |
| Storage temperature | –65°C to +150°C |
| Maximum junction temperature ($T_J$) | 175°C |
| Lead temperature (soldering, 10 sec max) | 300°C |
| Maximum current density $5x10^{-5}$ A/cm² per Mil-M-38510 | < $5x10^{-5}$ A/cm² |
| Maximum $\theta_{JC}$ = 28°C/W for cerdips per Mil-M-38510 | < 28°C/W |
| Maximum $\theta_{JC}$ = 22°C/W for flatpacks per Mil-M-38510 | < 22°C/W |
| Maximum $\theta_{JC}$ = 20°C/W for leadless chip carriers per Mil-M-38510 | < 20°C/W |

# Military Standard 24-Pin PAL Series

## PAL12L10, 14L8, 16L6, 18L4, 20L2, 20C1

Can be purchased to military drawing 5962-86804, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | –55 | | °C |
| $T_C$ | Operating case temperature | | 125 | °C |
| $V_{IL}$* | Low-level input voltage | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OS}$* | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 100 | mA |

*Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | 45 | ns |

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

# Military Decoder 24-Pin PAL Device

## PAL8L14A

### Operating Conditions

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | –55 | | °C |
| $T_C$ | Operating case temperature | | 125 | °C |
| $V_{IL}$* | Low-level input voltage | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$ | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$ | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2mA | 2.4 | | V |
| $I_{OS}$* | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 100 | mA |

*Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

5

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{PD}$ | Input to output propagation delay | $R_1$ = 560 Ω<br>$R_2$ = 1.1 KΩ | | 30 | ns |

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

# Military Very High Speed 24-Pin PAL Series

## PAL20L8B, 20R8B, 20R6B, 20R4B

Can be purchased to standard military drawing 5962-87671, latest revision in effect.

# Military High Speed 24-Pin PAL Series

## PAL20L8A, 20R8A, 20R6A, 20R4A

Can be purchased to standard military drawing 84129, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | 24 B | | 24 A | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | | 125 | °C |
| $t_w$† | Width of clock (except 20L8) | Low, $t_{wl}$ | 12 | | 20 | | ns |
| | | High, $t_{wh}$ | 12 | | 20 | | ns |
| $t_{su}$† | Set up time from input or feedback to clock (except 20L8) | | 20 | | 30 | | ns |
| $t_h$† | Hold time | | 0 | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = 4.5 V | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = 5.5 V | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = 5.5 V | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = 5.5 V | $V_I$ = 5.5 V | | 1.0 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = 4.5 V | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = 4.5 V | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Offstate output current | $V_{CC}$ = 5.5 V | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OHZ}$* | | | $V_O$ = 2.4 V | | 100 | |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5.5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = 5.5 V | | | 210 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Very High Speed 24-Pin PAL Series

## PAL20L8B, 20R8B, 20R6B, 20R4B

# Military High Speed 24-Pin PAL Series

## PAL20L8A, 20R8A, 20R6A, 20R4A

### Switching Characteristics Over operating conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | 24 B | | 24 A | | UNIT |
|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | |
| $t_{PD}$ | Input or feedback to output (except 20R8) | $R_1$ = 390 Ω<br>$R_2$ = 750 Ω | | 20 | | 30 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 20L8) | | | 15 | | 20 | ns |
| $t_{PZX}$ | Pin 13 to output enable (except 20L8) | | | 20 | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable (except 20L8) | | | 20 | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 20R8) | | | 25 | | 30 | ns |
| $t_{PXZ}$ | Input to output disable (except 20R8) | | | 20 | | 30 | ns |
| $f_{MAX}$* | State machine maximum operating frequency (except 20L8) | | 28.5 | | 20 | | MHz |
| | Data path register maximum operating frequency (except 20L8) | | 41.6 | | 25 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{su} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{su}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

# Military Half-Power 24A-Pin Series

## PAL20L8A-2, 20R8A-2, 20R6A-2, 20R4A-2

Can be purchased to standard military drawing 84129, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | 125 | °C |
| $t_w$† | Width of clock (except 20L8) | Low | 25 | | ns |
| | | High | 25 | | ns |
| $t_{su}$† | Setup time from input or feedback to clock (except 20L8) | | 50 | | ns |
| $t_h$† | Hold time | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Offstate output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply Current | $V_{CC}$ = MAX | | | 105 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military Half-Power 24A-Pin Series

## PAL20L8A-2, 20R8A-2, 20R6A-2, 20R4A-2

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output (except 20R8) | $R_1$ = 390 Ω, $R_2$ = 750 Ω | | 50 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 20L8) | | | 25 | ns |
| $t_{PZX}$ | Pin 13 to output enable (except 20L8) | | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable (except 20L8) | | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 20R8) | | | 45 | ns |
| $t_{PXZ}$ | Input to output disable (except 20R8) | | | 45 | ns |
| $f$MAX* | State machine maximum operating frequency (except 20L8) | | 13.3 | | MHz |
| | Data path register maximum operating frequency (except 20L8) | | 20 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

# Military High Speed 24XA-Pin Series

## PAL20L10A, 20X10A, 20X8A, 20X4A

Can be purchased to standard miltary print 84129, latest revision in effect.

### Operating Conditions

| SYMBOL | PARAMETER | | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | °C |
| $t_w$† | Width of clock (except 20L10) | Low | 35 | | ns |
| | | High | 20 | | |
| $t_{su}$† | Setup time from input or feedback to clock (except 20L10) | | 40 | | ns |
| $t_h$† | Hold time | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | 20X10A, 20X8A, 20X4A | | 180 | mA |
| | | | 20L10A | | 165 | |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military High Speed 24XA-Pin Series

## PAL20L10A, 20X10A, 20X8A, 20X4A

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $t_{PD}$ | 20L10A, 20X8A, 20X4A<br>Input or feedback to output (except 20X10) | $R_1$ = 390 Ω<br>$R_2$ = 750 Ω | | 35 | ns |
| $t_{CLK}$ | Clock to output or feedback (except 20L10) | | | 25 | ns |
| $t_{PZX}$ | Pin 13 to output enable (except 20L10) | | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable (except 20L10) | | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 20X10) | | | 35 | ns |
| $t_{PXZ}$ | Input to output disable (except 20X10) | | | 35 | ns |
| *f*MAX* | State machine maximum operating frequency (except 20L10) | | 15.4 | | MHz |
| | Data path register maximum operating frequency (except 20L10) | | 18.2 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{su} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{su}+t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

# Military 24RS-Pin Series

## PAL20S10, 20RS10, 20RS8, 20RS4

Standard military 5962-87530 is in the process of being generated—Contact the factory.

### Operating Conditions

| SYMBOL | PARAMETER | | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | °C |
| $t_w$† | Width of clock (except 20S10) | Low | 20 | | ns |
| | | High | 20 | | |
| $t_{su}$† | Setup time from input or feedback to clock (except 20S10) | | 40 | | ns |
| $t_h$† | Hold time | | 0 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 240 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military 24RS-Pin Series

## PAL20S10, 20RS10, 20RS8, 20RS4

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output (except 20RS10) | Polarity fuse intact | | | 40 | ns |
| | | Polarity fuse Blown | | | 45 | |
| $t_{CLK}$ | Clock to output or feedback (except 20S10) | | | | 20 | ns |
| $t_{PZX}$ | Pin 13 to output enable (except 20S10) | | $R_1$ = 390 Ω | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable (except 20S10) | | $R_2$ = 750 Ω | | 25 | ns |
| $t_{PZX}$ | Input to output enable (except 20RS10) | | | | 35 | ns |
| $t_{PXZ}$ | Input to output disable (except 20RS10) | | | | 30 | ns |
| $f$MAX* | State machine maximum operating frequency (except 20S10) | | | 16.7 | | MHz |
| | Data path register maximum frequency (except 20S10) | | | 25 | | |

*$f$MAX is calculated and measured on initial qualifications only.

$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$

$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU} + t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

# Military 24RA-Pin Device

## PAL20RA10

Standard military 5962-86803 is in the process of being generated—Contact the factory.

### Operating Conditions

| SYMBOL | PARAMETER | | MIN | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | | °C |
| $T_C$ | Operating case temperature | | | 125 | °C |
| $t_w$† | Width of clock | Low | 25 | | ns |
| | | High | 25 | | |
| $t_{wp}$† | Preload pulse width | | 45 | | ns |
| $t_{su}$† | Setup time from input or feedback to clock | | 25 | | ns |
| $t_{sup}$† | Preload setup time | | 30 | | ns |
| $t_h$† | Hold time | Polarity fuse intact | 10 | | ns |
| | | Polarity fuse blown | 0 | | |
| $t_{hp}$† | Preload hold time | | 30 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.

* These are absolute voltages with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.

† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | | 1 | mA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 8 mA | | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | | 100 | |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | | 200 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.

** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

# Military 24RA-Pin Device

## PAL20RA10

### Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITIONS | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | Polarity fuse intact | | | 35 | ns |
| | | Polarity fuse Blown | | | 40 | |
| $t_{CLK}$ | Clock to output or feedback | | | | 35 | ns |
| $t_S$ | Input to asynchronous set | | | | 40 | ns |
| $t_R$ | Input to asychronous reset | | $R_1$ = 560 Ω | | 45 | ns |
| $t_{PZX}$ | Pin 13 to output enable | | $R_2$ = 1.1 KΩ | | 25 | ns |
| $t_{PXZ}$ | Pin 13 to output disable | | | | 25 | ns |
| $t_{PZX}$ | Input to output enable | | | | 35 | ns |
| $t_{PXZ}$ | Input to output disable | | | | 35 | ns |
| $f$MAX* | State machine maximum operating frequency | | | 16.7 | | MHz |
| | Data path register maximum frequency | | | 20 | | |

*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (state machine) = $1/[t_{SU} + t_{CLK}]$
$f$MAX (data path register) = $1/[t_{WL} + t_{WH}]$ or $1/t_{SU} + t_h$, whichever is smaller.

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

## ADVANCE INFORMATION

### Operating Conditions

| SYMBOL | PARAMETER | | STD MIN | STD MAX | A MIN | A MAX | UNIT |
|---|---|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | 4.5 | 5.5 | 4.5 | 5.5 | V |
| $T_A$ | Operating free-air temperature | | –55 | 125 | –55 | 125 | °C |
| $t_w$† | Width of clock | Low | 25 | | 23 | | ns |
| | | High | 25 | | 23 | | |
| $t_{su}$† | Setup time from input or feedback to clock | Product terms $P_1$–$P_n$, SR | 35 | | 30 | | ns |
| | | Product term XOR | 40 | | 35 | | |
| $t_h$† | Hold time | | 0 | | 0 | | ns |
| $t_{aw}$† | Asynchronous preset width | | 35 | | 30 | | ns |
| $t_{ar}$† | Asynchronous preset recovery time | | 35 | | 30 | | ns |
| $t_{sr}$† | Synchronous reset recovery time | | 35 | | 30 | | ns |
| $V_{IL}$* | Low-level input voltage | | | ≤0.8 | | ≤0.8 | V |
| $V_{IH}$* | High-level input voltage | | ≥2.0 | | ≥2.0 | | V |

Note: Virgin array verify of unprogrammed PAL device is performed at 25°C only.
* These voltages apply with respect to the ground pin on the device and include all overshoots due to system and/or tester noise.
† These are device set-up conditions, which are measured during initial qualification, and are not directly tested.

### Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | TEST CONDITIONS | | MIN | MAX | UNIT |
|---|---|---|---|---|---|---|
| $V_{IC}$ | Input clamp voltage | $V_{CC}$ = MIN | $I_I$ = –18 mA | | –1.5 | V |
| $I_{IL}$* | Low-level input current | $V_{CC}$ = MAX | $V_I$ = 0.4 V | | –0.25 | mA |
| $I_{IH}$* | High-level input current | $V_{CC}$ = MAX | $V_I$ = 2.4 V | | 25 | µA |
| $I_I$ | Maximum input current | $V_{CC}$ = MAX | $V_I$ = 5.5 V | | 200 | µA |
| $V_{OL}$ | Low-level output voltage | $V_{CC}$ = MIN | $I_{OL}$ = 12 mA | | 0.5 | V |
| $V_{OH}$ | High-level output voltage | $V_{CC}$ = MIN | $I_{OH}$ = –2 mA | 2.4 | | V |
| $I_{OZL}$* | Off-state output current | $V_{CC}$ = MAX | $V_O$ = 0.4 V | | –100 | µA |
| $I_{OZH}$* | | | $V_O$ = 2.4 V | | 100 | µA |
| $I_{OS}$** | Output short-circuit current | $V_{CC}$ = 5 V | $V_O$ = 0.5 V | –30 | –130 | mA |
| $I_{CC}$ | Supply current | $V_{CC}$ = MAX | | | 180 | mA |

* I/O pin leakage is worst case of IIX or IOZX; i.e., IIL and IOZH.
** Not more than one output should be shorted at a time and duration of the short circuit should not exceed one second.

## Switching Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | | TEST CONDITIONS | STD MIN | STD MAX | A MIN | A MAX | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| $t_{PD}$ | Input or feedback to output | | Product terms $P_1$–$P_n$ | $R_1$ = 390 Ω, $R_2$ = 750 Ω | | 35 | | 30 | ns |
| | | | Product term XOR | | | 40 | | 35 | |
| $t_{CLK}$ | Clock to output or feedback | | | | | 20 | | 20 | ns |
| $t_{PZX}$ | Input to output enable | | | | | 35 | | 30 | ns |
| $t_{PXZ}$ | Input to output disable | | | | | 35 | | 30 | ns |
| $t_{AP}$ | Asynchronous preset to output | | | | | 35 | | 30 | ns |
| $t_{CR}$ | Input or feedback to registered output from combinatorial configuration | | | | | 95 | | 95 | ns |
| $t_{RC}$ | Input or feedback to combinatorial output from registered configuration | | | | | 95 | | 95 | ns |
| $f$MAX* | Maximum frequency | Feedback ($1/t_{P1}$) | Product terms $P_1$–$P_n$ | | 18 | | 20 | | MHz |
| | | | Product term XOR | | 16.7 | | 18 | | |
| | | No feedback** | | | 20 | | 21.7 | | |

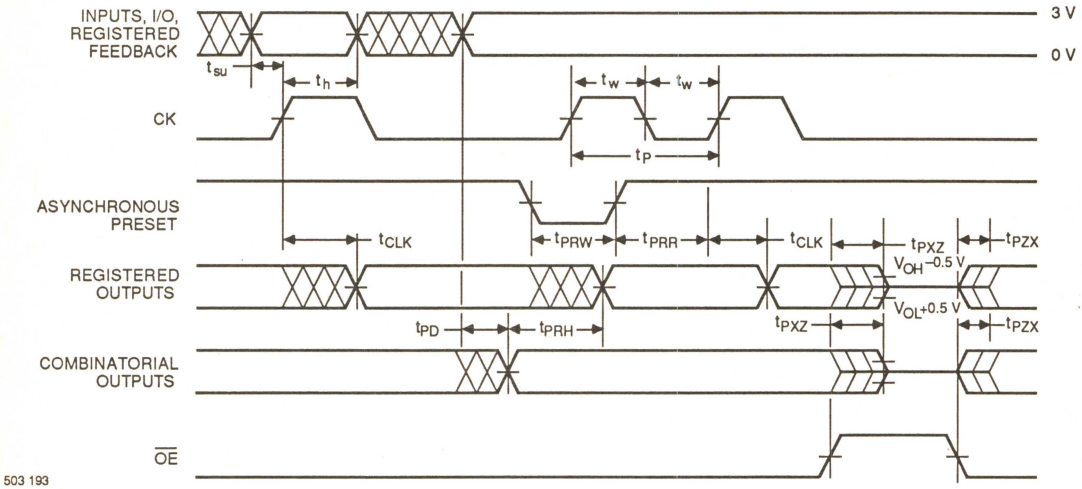
*$f$MAX is calculated and measured on initial qualifications only.
$f$MAX (NO feedback) = $1/[t_{WL} + t_{WH}]$

Programmed devices conform to Mil-Std-883, Method 5005, Group A, Subgroups 1, 2, 3, 7, 8, 9, 10 and 11.

5

# Military 24-Pin PAL Devices

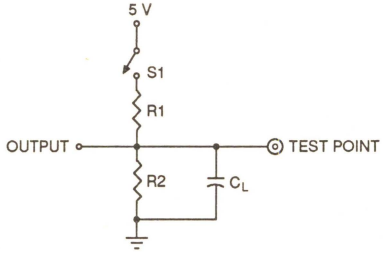
## Switching Waveforms




503 193

Notes:
1. $t_{PD}$ is tested with switch $S_1$ closed. $C_L$ = 50 pF and measured at 1.5 V output level.
2. $t_{PZX}$ is measured at the 1.5 V output level with $C_L$ = 50 pF. $S_1$ is open for high impedance to "1" test and closed for high impedance "0" test.
3. $t_{PXZ}$ is tested with $C_L$ = 5 pF. $S_1$ is open for "1" to high impedance test measured at $V_{OH}$ −0.5 V output level. $S_1$ is closed for "0" to high impedance test measured at $V_{OL}$ +0.5 V output level.
4. Equivalent test loads may be used on automatic test equipment.

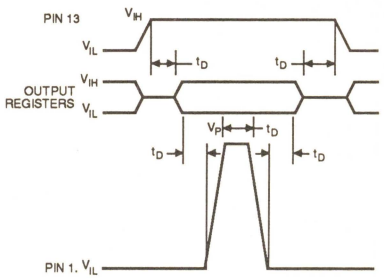
## Switching Test Load




503 194

## Output Register Preload PAL24XA Series, PAL24RS Series and PAL20RA10 Device

1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 13 to $V_{IH}$.
3. Apply $V_{IL}/V_{IH}$ to all registered output pins.
4. Pulse pin 10 to Vp then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all output registers.
6. Lower pin 13 to $V_{IL}$ to enable the output registers.
7. Verify for $V_{OL}/V_{OH}$ at all registered output pins.

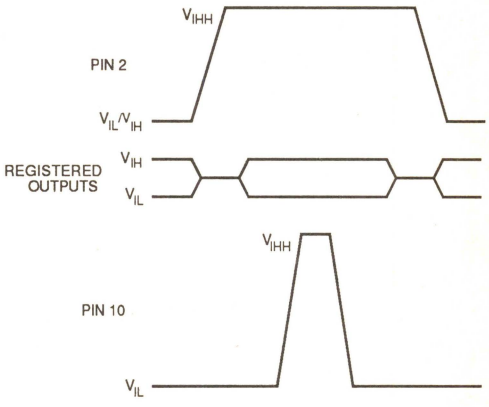



503 195

## Output Register Preload PAL32VX10 Device

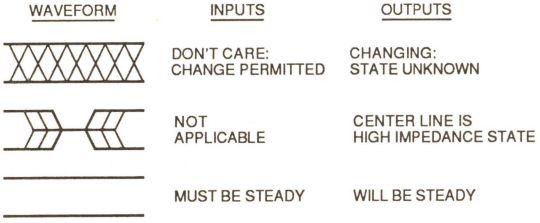
The preload function allows the register to be loaded from the output pins. This feature aids the functional testing of sequential designs by allowing direct setting of output states. The procedure is:

1. Raise $V_{CC}$ to 4.5 V.
2. Disable output registers by setting pin 2 to $V_{IHH}$(12 V).
3. Apply $V_{IL}/V_{IH}$ to all registered output pins. Leave combinatorial outputs floating.
4. Pulse pin 10 to $V_{IHH}$, then back to 0 V.
5. Remove $V_{IL}/V_{IH}$ from all output registers.
6. Remove high voltage from pin 2.
7. Enable registered outputs per programmed pattern.
8. Verify for $V_{OL}/V_{OH}$ at all registered output pins.




503 196

## Key to Timing Diagrams




503 197

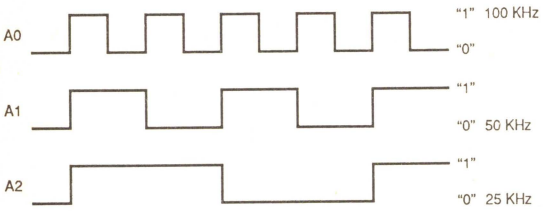
# Military 24-Pin PAL Devices

## Life Test/Burn-In Circuits

Complies with Mil-Std-883, Method 1005/1015, Condition D.

## Circuit Configurations

## Waveforms

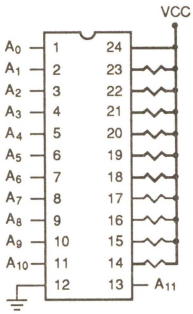



A3 to Ax Follow in order

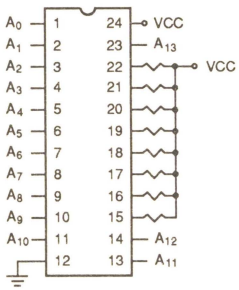
503 134

1. All Burn-In will be accomplished at 125°C +5/–0°C
2. $V_{CC}$ = 5.25 volts ± 0.25 V
3. All Clocks (A0 to Ax) are square wave signals 50±15% Duty Cycle, with:
   a. "0" = –0.5 V to +0.7 V
   b. "1" = +2.4 V to $V_{CC}$
   c. Rise Time (+0.7 V to + 2.4 V) < 1 µsec
   d. Fall Time (+2.4 V to +0.7 V) < 1 µsec
4. Resistor Value
   330 Ω or 470 Ω ±5%
5. All Board Components to be compatible with 150°C Ambient (Min).

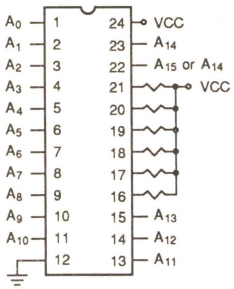
**PAL12L10**




**PAL14L8**




**PAL16L6**




503 199

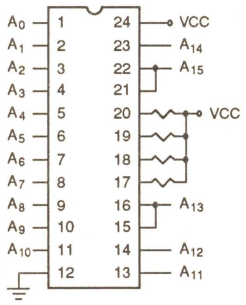
# Military 24-Pin PAL Devices

## Life Test/Burn-In Circuits

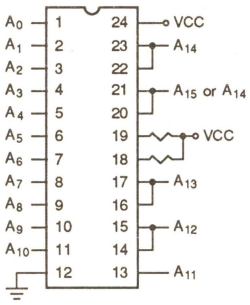
Complies with Mil-Std-883, Method 1005/1015, Condition D.

## Circuit Configurations

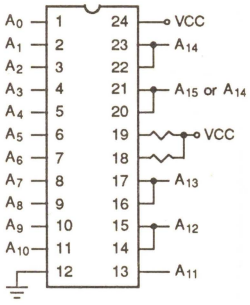
### PAL18L4



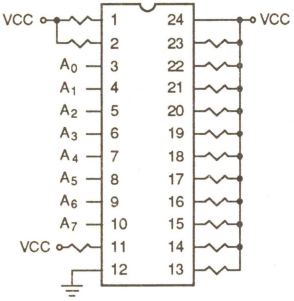

### PAL20L2



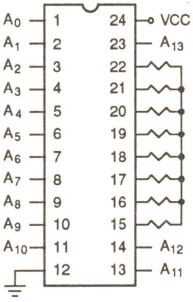

### PAL20C1



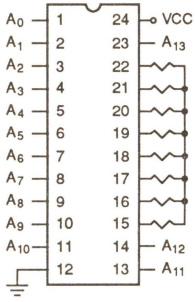

### PAL8L14A



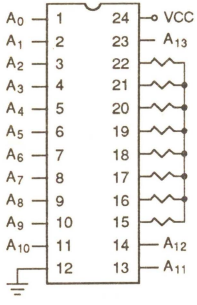

### PAL20L8A/B PAL20L8A-2



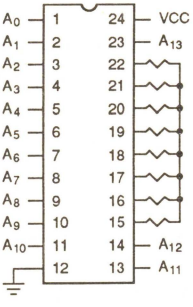

### PAL20R8A/B PAL20R8A-2



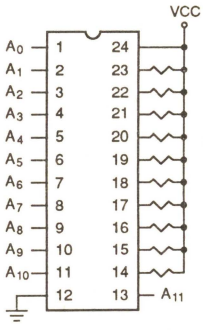

### PAL20R6A/B PAL20R6A-2




### PAL20R4A/B PAL20R4A-2




### PAL20L10A




503 202

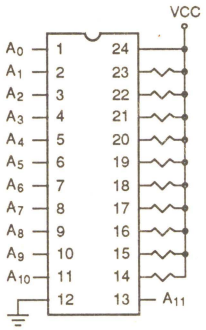
# Military 24-Pin PAL Devices

## Life Test/Burn-In Circuits

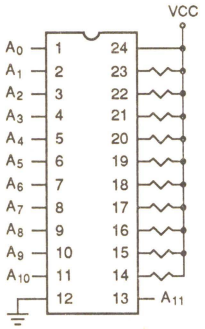
Complies with Mil-Std-883, Method 1005/1015, Condition D.

## Circuit Configurations

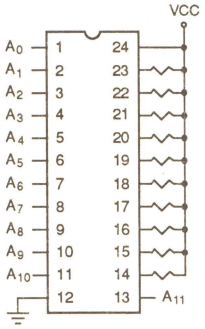
**PAL20X10A**



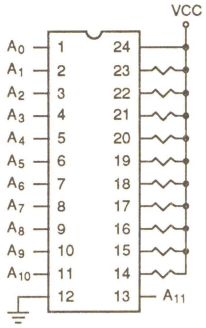

**PAL20X8A**



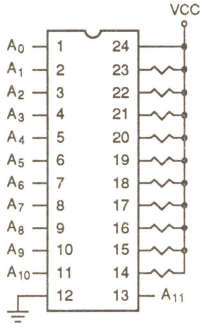

**PAL20X4A**



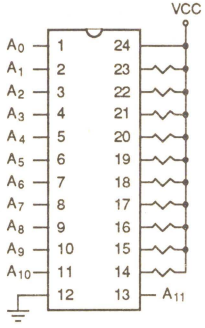

**PAL20S10**



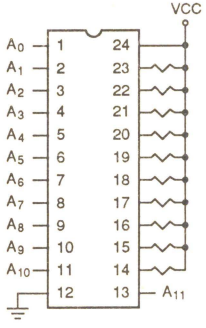

**PAL20RS10**



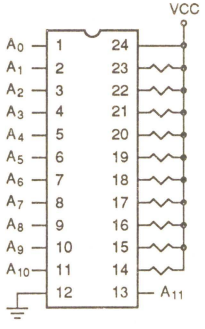

**PAL20RS8**




**PAL20RS4**




**PAL32VX10/10A**




503 211

## Programming Inputs for ProPAL and HAL Devices

1. • Mag Tape* and one master
   • Floppy Disk* and one master
   • 2 masters and equation printout or truthtable
   • 2 masters alone (requires customer approval)
   • Stand alone equations (requires customer approval)

*2. The following is a list describing the different types of Mag Tapes and Floppy Disks that Software Support currently can accept:
   • Standard 8 Inch Floppy Disks formatted RX01/IBM0 (Single-Sided, Single Density) or RX02/IBM2 (Single-Sided, Double Density) or RX03/IBM3 (Double-Sided, Double Density).
   • IBM 51/4 Inch Floppy Disks formatted Single-Sided, Double Density or Double-Sided, Double Density
   • Magnetic Tape (Created on VAX/VMS System): Mag Tapes must be in IBM compatible (800 or 1600 BPI) nine track in blocked, unlabeled (card image) format of Files-11 or VAX/VMS Backup format.

   or

   • Magnetic Tape (Created on UNIX system for reading on VAX/VMS system, 800 or 1600 BPI): Write using 'dd' with block size 150. MMI will, provide a UNIX shell program upon request for writing multiple files to a tape with the correct parameters.

   or (last resort)

   • Magnetic Tape (Created on UNIX system for reading on UNIX system, 1600 or 3200 BPI): Write using 'tar-cv' or similar command.
   • In all cases, the Tapes or Disks must contain a label indicating all data such as the density, the format, the operating system, the command used to write the files and/or to remove the data from the Tape or Disk, and a listing of the filenames.

3. For programmed PAL devices, the customer must provide a design package including Boolean Equations, "Seed" function test sequence, package stipulation and AC test vectors, when required. Delivery quotes for this type of product begin after receipt of this data from the customer.

4. To give you an idea of the delivery differences for the options discussed in DC/AC Parametric Testing, general lead times are as follows:
   • Unprogrammed:
     Cerdip, 4–6 weeks
     Cerpack/Flatpack, 8–12 weeks
     Leadless Chip Carriers 6–12 weeks
   • (Option 1) Unprogrammed product using our standard pattern to verify AC at room temperature on sample basis. Add 2 weeks to standard delivery.
   • (Option 2) Programmed product using customer patterns. Contact factory for delivery. Delivery quoted will be after receipt of customer design package.

## DC/AC Parametric Testing

### 1. VIL/VIH Parametric Information

VIL/VIH parameters are, in effect, input conditions of DC tests and are not directly tested. Functional tests are performed at VIL = 0.4 V and VIH = 2.4 V. VIL is specified at ≤ 0.8 V, and VIH is specified at ≥ 2.0 V.

### 2. AC Testing/Programming

Although Monolithic Memories offers a large selection of programmable products, it must be pointed out that AC Testing cannot be performed on many of our products types without their being programmed. For those devices which must be programmed prior to AC Tests and are ordered unprogrammed, Monolithic Memories must "guarantee" their AC Performance.

Since the guaranteeing of parameters can be a serious concern for the Military user, we have outlined several approaches to address the AC screening issue.

Option 1. Monolithic Memories can pull a Sample from a lot using our own Standard patterns (designed to blow in excess of 50 percent of the fuses) and perform AC testing at 25°C.
   • PAL products processed to Military drawings include programmability samples and AC testing at 25°C.

Option 2. Monolithic Memories can program PROMs using custom patterns submitted by the customer. AC testing can then be done with the following options:
   • Sample AC at 25°C
   • Sample AC at 25°C, –55°C, 125°C
   • 100% AC at 25°C
   • 100% AC at 25°C, –55°C and 125°C

Note: For PAL devices contact the factory.

On PAL products where custom programming is performed and AC testing is required, additional vector generation and fault coverage analysis is required, as well as AC program generation and checkout.

## JAN 38510 and Standard Military Drawing Program

Advanced Micro Devices and Monolithic Memories are active participants in the JAN 38510 and Standard Military Drawing (SMD) Program. The idea behind the SMD Program is to standardize MIL-STD-883, Class B microcircuits where fully qualified JAN product is not available. The advantage to the user is that SMDs are a cost effective alternative to source control drawings and are offered as off-the-shelf stocking items by IC manufacturers participating in the program.

Standard Military Drawings should always be considered to improve availability over source control drawings. It is standard practice at AMD/MMI to convert our 883, Class B processing to SMDs for all products which we are approved to supply. AMD/MMI then dual marks these devices with both the SMD number and the Generic Part Number. DESC approved products can then be procured to either part number as standard product through both OEM and Distributor channels.

The following cross reference will allow you to determine the appropriate SMD and JAN Drawing for each PAL device. AMD/MMI will continue to work closely with DESC, generating new drawings, which will provide a steady flow of advanced technology products to standardized specifications.

## MIL-M-38510 Slash Sheet Cross Reference for AMD / MMI Generic Part Number

| M38510 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 503 | 10H8 | 12H6 | 14H4 | | | 10L8 | 12L6 | 14L4 | | |
| 504 | 16L8A | 16R8A | 16R6A | 16R4A | | | 16L8A-2 | 16R8A-2 | 16R4A-2 | 16R4A-2 |
| 505 | 20L8A | 20R8A | 20R6A | 20R4A | | | | | | |

## Standard Military Drawing Generic Part Type Cross Reference

| STANDARD MILITARY DRAWING PART NUMBER | GENERIC PART NUMBER | JAN REPLACEMENT PART NUMBER |
|---|---|---|
| 8103501RA | PAL10H8MJ/883B | M38510/50301BRA |
| 81035012A | PAL10H8ML/883B | - |
| 8103501SA | PAL10H8MW/883B | - |
| 8103502RA | PAL12H6MJ/883B | M38510/50302BRA |
| 81035022A | PAL12H6ML/883B | - |
| 8103502SA | PAL12H6MW/883B | - |
| 8103503RA | PAL14H4MJ/883B | M38510/50303BRA |
| 81035032A | PAL14H4ML/883B | - |
| 8103503SA | PAL14H4MW/883B | - |
| 8103504RA | PAL16H2MJ/883B | - |
| 81035042A | PAL16H2ML/883B | - |
| 8103504SA | PAL16H2MW/883B | - |
| 8103505RA | PAL16C1MJ/883B | - |
| 81035052A | PAL16C1ML/883B | - |
| 8103505SA | PAL16C1MW/883B | - |
| 8103506RA | PAL10L8MJ/883B | M38510/50306BRA |
| 81035062A | PAL10L8ML/883B | - |
| 8103506SA | PAL10L8MW/883B | - |
| 8103507RA | PAL12L6MJ/883B | M38510/50307BRA |
| 81035072A | PAL12L6ML/883B | - |
| 8103507SA | PAL12L6MW/883B | - |
| 8103508RA | PAL14L4MJ/883B | M38510/50308BRA |
| 81035082A | PAL14L4ML/883B | - |
| 8103508SA | PAL14L4MW/883B | - |
| 8103509RA | PAL16L2MJ/883B | - |
| 81035092A | PAL16L2ML/883B | - |
| 8103509SA | PAL16L2MW/883B | - |

## Standard Military Drawing Generic Part Type Cross Reference (Cont'd.)

| STANDARD MILITARY DRAWING PART NUMBER | GENERIC PART NUMBER | JAN REPLACEMENT PART NUMBER |
|---|---|---|
| 8103607RA | PAL16L8AMJ/883B | M38510/50401BRA |
| 81036072A | PAL16L8AML/883B | - |
| 8103607SA | PAL16L8AMW/883B | - |
| 8103608RA | PAL16R8AMJ/883B | M38510/50402BRA |
| 81036082A | PAL16R8AML/883B | - |
| 8103608SA | PAL16R8AMW/883B | - |
| 8103609RA | PAL16R6AMJ/883B | M38510/50403BRA |
| 81036092A | PAL16R6AML/883B | - |
| 8103609SA | PAL16R6AMW/883B | - |
| 8103610RA | PAL16R4AMJ/883B | M38510/50404BRA |
| 81036102A | PAL16R4AML/883B | - |
| 8103610SA | PAL16R4AMW/883B | - |
| 8103611RA | PAL16L8A-2MJ/883B | M38510/50407BRA |
| 81036112A | PAL16L8A-2ML/883B | M38510/50407B2A |
| 8103611SA | PAL16L8A-2MW/883B | - |
| 8103612RA | PAL16R8A-2MJ/883B | M38510/50408BRA |
| 81036122A | PAL16R8A-2ML/883B | M38510/50408B2A |
| 8103612SA | PAL16R8A-2MW/883B | - |
| 8103613RA | PAL16R6A-2MJ/883B | M38510/50409BRA |
| 81036132A | PAL16R6A-2ML/883B | M38510/50409B2A |
| 8103613SA | PAL16R6A-2MW/883B | - |
| 8103614RA | PAL16R4A-2MJ/883B | M38510/50410BRA |
| 81036142A | PAL16R4A-2ML/883B | M38510/50410B2A |
| 8103614SA | PAL16R4A-2MW/883B | - |
| | | |
| 8412901LA | PAL20L8AMJS/883B | M38510/50501BLA |
| 84129013A | PAL20L8AML/883B | M38510/50501B3A |
| 8412901KA | PAL20L8AMW/883B | - |
| 8412902LA | PAL20R8AMJS/883B | M38510/50502BLA |
| 84129023A | PAL20R8AML/883B | M38510/50502B3A |
| 8412902KA | PAL20R8AMW/883B | - |
| 8412903LA | PAL20R6AMJS/883B | M38510/50503BLA |
| 84129033A | PAL20R6AML/883B | M38510/50503B3A |
| 8412903KA | PAL20R6AMW/883B | - |
| 8412904LA | PAL20R4AMJS/883B | M38510/50504BLA |
| 84129043A | PAL20R4AML/883B | M38510/50504B3A |
| 8412904KA | PAL20R4AMW/883B | - |
| 8412905LA | PAL20L10AMJS/883B | - |
| 84129053A | PAL20L10AML/883B | - |
| 8412905KA | PAL20L10AMW/883B | - |
| 8412906LA | PAL20X8AMJS/883B | - |
| 84129063A | PAL20X8AML/883B | - |
| 8412906KA | PAL20X8AMW/883B | - |
| 8412907LA | PAL20X10AMJS/883B | - |
| 84129073A | PAL20X10AML/883B | - |
| 8412907KA | PAL20X10AMW/883B | - |
| 8412908LA | PAL20X4AMJS/883B | - |
| 84129083A | PAL20X4AML/883B | - |
| 8412908KA | PAL20X4AMW/883B | - |
| 8412909LA | PAL20L8A-2MJS/883B | - |
| 84129093A | PAL20L8A-2ML/883B | - |
| 8412909KA | PAL20L8A-2MW/883B | - |
| 8412910LA | PAL20R8A-2MJS/883B | - |
| 84129103A | PAL20R8A-2ML/883B | - |
| 8412910KA | PAL20R8A-2MW/883B | - |
| 8412911LA | PAL20R6A-2MJS/883B | - |

## Standard Military Drawing Generic Part Type Cross Reference (Cont'd.)

| STANDARD MILITARY DRAWING PART NUMBER | GENERIC PART NUMBER | JAN REPLACEMENT PART NUMBER |
|---|---|---|
| 84129113A | PAL20R6A-2ML/883B | - |
| 8412911KA | PAL20R6A-2MW/883B | - |
| 8412912LA | PAL20R4A-2MJS/883B | - |
| 84129123A | PAL20R4A-2ML/883B | - |
| 8412912KA | PAL20R4A-2MW/883B | - |
| | | |
| 8506501RA | PAL16L8A-4MJ/883B | - |
| 85065012A | PAL16L8A-4ML/883B | - |
| 8506501SA | PAL16L8A-4MW/883B | - |
| 8506502RA | PAL16R8A-4MJ/883B | - |
| 85065022A | PAL16R8A-4ML/883B | - |
| 8506502SA | PAL16R8A-4MW/883B | - |
| 8506503RA | PAL16R6A-4MJ/883B | - |
| 85065032A | PAL16R6A-4ML/883B | - |
| 8506503SA | PAL16R6A-4MW/883B | - |
| 8506504RA | PAL16R4A-4MJ/883B | - |
| 85065042A | PAL16R4A-4ML/883B | - |
| 8506504SA | PAL16R4A-4MW/883B | - |
| | | |
| 5962-8515501RA | PAL16L8BMJ/883B | - |
| 5962-85155012A | PAL16L8BML/883B | - |
| 5962-8515501SA | PAL16L8BMW/883B | - |
| 5962-8515502RA | PAL16R8BMJ/883B | - |
| 5962-85155022A | PAL16R8BML/883B | - |
| 5962-8515502SA | PAL16R8BMW/883B | - |
| 5962-8515503RA | PAL16R6BMJ/883B | - |
| 5962-85155032A | PAL16R6BML/883B | - |
| 5962-8515503SA | PAL16R6BMW/883B | - |
| 5962-8515504RA | PAL16R4BMJ/883B | - |
| 5962-85155042A | PAL16R4BML/883B | - |
| 5962-8515504SA | PAL16R4BMW/883B | - |
| 5962-8515505RA | PAL16L8B-2MJ/883B | - |
| 5962-85155052A | PAL16L8B-2ML/883B | - |
| 5962-8515505SA | PAL16L8B-2MW/883B | - |
| 5962-8515506RA | PAL16R8B-2MJ/883B | - |
| 5962-85155062A | PAL16R8B-2ML/883B | - |
| 5962-8515506SA | PAL16R8B-2MW/883B | - |
| 5962-8515507RA | PAL16R6B-2MJ/883B | - |
| 5962-85155072A | PAL16R6B-2ML/883B | - |
| 5962-8515507SA | PAL16R6B-2MW/883B | - |
| 5962-8515508RA | PAL16R4B-2MJ/883B | - |
| 5962-85155082A | PAL16R4B-2ML/883B | - |
| 5962-8515508SA | PAL16R4B-2MW/883B | - |
| 5962-8515509RA | PAL16L8DMJ/883B | - |
| 5962-85155092A | PAL16L8DML/883B | - |
| 5962-8515509SA | PAL16L8DMW/883B | - |
| 5962-8515510RA | PAL16R8DMJ/883B | - |
| 5962-85155102A | PAL16R8DML/883B | - |
| 5962-8515510SA | PAL16R8DMW/883B | - |
| 5962-8515511RA | PAL16R6DMJ/883B | - |
| 5962-85155112A | PAL16R6DML/883B | - |
| 5962-8515511SA | PAL16R6DMW/883B | - |
| 5962-8515512RA | PAL16R4DMJ/883B | - |
| 5962-85155122A | PAL16R4DML/883B | - |
| 5962-8515512SA | PAL16R4DMW/883B | - |
| 5962-8680301LA | PAL20RA10MJS/883B | - |

## Standard Military Drawing Generic Part Type Cross Reference (Cont'd.)

| STANDARD MILITARY DRAWING PART NUMBER | GENERIC PART NUMBER | JAN REPLACEMENT PART NUMBER |
|---|---|---|
| 5962-8680301KA | PAL20RA10MW/883B | - |
| 5962-86803013A | PAL20RA10ML/883B | - |
| 5962-8752801RA | AmPAL18P8A/BRA | - |
| 5962-87528012A | AmPAL18P8A/B2A | - |
| 5962-8752801SA | AmPAL18P8A/BSA | - |
| 5962-8752802RA | AmPAL18P8B/BRA | - |
| 5962-87528022A | AmPAL18P8B/B2A | - |
| 5962-8752802SA | AmPAL18P8B/VSA | - |
| 5962-8752803RA | AmPAL18P8AL/BRA | - |
| 5962-87528032A | AmPAL18P8AL/B2A | - |
| 5962-8752803SA | AmPAL18P8AL/BSA | - |
| 5962-8752804RA | AmPAL18P8L/BRA | - |
| 5962-87528042A | AmPAL18P8L/B2A | - |
| 5962-8752804SA | AmPAL18P8L/BSA | - |
| 5962-8605301LA | AmPAL22V10A/BLA | - |
| 5962-86053013A | AmPAL22V10A/B3A | - |
| 5962-9605301KA | AmPAL22V10A/BKA | - |
| 5962-8605302LA | AmPAL22V10/BLA | - |
| 5962-86053023A | AmPAL22V10/B3A | - |
| 5962-8605302KA | AmPAL22V10/BKA | - |

| MMI PACKAGE DESIGNATOR | PACKAGE TYPE | STANDARD LEAD FINISH | MIL-SPEC LEAD FINISH DESIGNATOR |
|---|---|---|---|
| J/JS | CERAMIC DIP | SOLDER DIP | A |
| W | CERAMIC FLATPACK | SOLDER DIP | A |
| L | CERAMIC LEADLESS CHIP CARRIER | SOLDER DIP | A |

## Product Introduction Procedures

All new products released by the Military Products Division must successfully pass Mil-Std-883 Class B processing prior to new product announcement. This practice allows us to do checkout of bonding diagrams, electrical test tapes and burn circuits in a manufacturing environment. Programmability is checked when applicable. Our Military Engineering Department reviews electrical data to insure performance and yields to military data sheet limits are acceptable, prior to new product release. This procedure allows MPD to keep manufacturing start-up problems to a minimum on new product orders.

## Standard Processing Flows

Monolithic Memories Processing and Screening flows are organized to provide a broad selection of processing options, structured around the most commonly requested customer flows.

Standard processing flows for the Military Products Division include:

Monolithic Memories Inc. Modified Level S
JAN 38510 Class B
Military Drawing Program
Mil-Std-883 Class B
Monolithic Memories Inc. Mil-Temp Product

In addition, these flows are expanded to provide for factory programming on PAL circuits, when required by our customers.

Major benefits can be realized by ordering product to standard flows whenever possible:

- Minimize need for source control drawings.
- Cost savings on unit cost—no price adders for custom processing.
- Improved lead time—no spec review or negotiation time, plus the ability to pull product from various work-in-process stages or purchase product from finished goods inventory.

For your reference, we have included our Modified Level S flow, our Mil-Std-883 Class B flow and our Mil-Temp Product flow. For your planning purposes, we have calculated typical throughput times for each operation, as product proceeds through the processing flow.

It is the policy of Monolithic Memories, to always operate to the most current revision of Mil-M-38510 and Mil-Std-883.

## Manufacturing and Screening Locations

JAN Products, Monolithic Memories Modified Level "S", and customer orders which call for U.S.A. assembly, are manufactured in our DESC certified line in Sunnyvale, California.

MIL-STD-883 Class B products, and orders to source control drawings, where stateside build is not required, are assembled at our Penang, Malaysia facility. This facility is qualified by Monolithic Memories Quality Department, as well as by many of our customers, to manufacture MIL-STD-883 Class B product. Conformance to MIL-STD-883 requirements is routinely monitored through audits at the Penang facility as well as incoming inspections in Sunnyvale. Manufacturing capabilities for each Monolithic Memories facility are highlighted on the chart below.

To identify the assembly location of each military device, the Country of origin is marked on all products prior to shipment. Products assembled in our stateside facility in Sunnyvale, California, will have "USA" marked on the topside of the device. The exception to this is JAN 38510 product, which is marked to the MIL-M-38510 requirements only.

Offshore built product, which is manufactured in Penang, Malaysia, will have "Malaysia" or "Malay" marked on the bottom side of the device.

## Manufacturing Capabilities

| | SUNNYVALE | PENANG |
|---|---|---|
| Assembly | X | X |
| Precap Inspection | X | X |
| Environmental Testing | X | X |
| Electrical Pre-Test | X | X |
| Burn-In | X | X |
| Post Burn–In Electricals | X | X |
| Group A Testing | X | X |
| Mark | X | X |
| Factory Programming | X | |
| Qualification and Quality Conformance Testing | X | |

## Standard Military Flow Chart

| SCREENING | MODIFIED LEVEL S | REQUIREMENT | CLASS B | REQUIREMENT |
|---|---|---|---|---|
| | MIL-STD-883 METHOD 5004 | | MIL-STD-883 METHOD 5004 | |
| S.E.M. | 2018 | Sample | | |
| Assembly | USAAssembly | | Typically offshore assembly | |
| Non-destruct bond pull | 2023(Sample) | LTPD = 5<br>REJ = 0<br>SS = 2 all wires | | |
| Die shear/ Destruct bond pull | 2019 (sample) | SS = 2<br>REJ = 0 | | |
| Internal visual | 2010 cond. A (modified) | 100% | 2010 cond. B | 100% |
| Stabilization bake | 1008 | 100% | 1008 | 100% |
| Temperature cycling | 1010 | 100% | 1010 | 100% |
| Constant acceleration | 2001 test cond. D or E Y1 orientation only | 100% | 2001 test cond. D or E Y1 orientation only | 100% |
| Seal A) Fine<br>B) Gross | | | 1014 cond. A or B cond. C | 100% |
| Particle impact noise detection (PIND) | 2020 cond. A only | 100% | | |
| Interim electrical parameters | Per application device specification<br>TA = 25°C only | 100% | | |
| Serialization | | 100% | | |
| X-Ray | 2010 two views<br>X and Y axis only | 100% | | |
| Interim electrical (1) parameter | Per applicable device specification<br>TA = 25°C only | 100% | Per applicable device (1) specification<br>TA = 25°C only | 100% |
| Post electrical parameters | Per applicable device specification<br>TA = 25°C only<br>(delta's when required) | 100% | Per applicable device specification<br>TA = 25°C only | 100% |
| Delta calculations (when applicable) | Per applicable device specification | | | |
| Percent defect allowable | DC Parameters<br>PDA = 5% or 1 device whichever is greater<br>Functional Parameters<br>PDA = 3% or 1 device whichever is greater | | DC Parameters<br>PDA = 5% or 1 device whichever is greater | |

(1) Programming and verification are performed at 25°C only.
(2) Unprogrammable PAL Devices–AC parameters are tested on programmed sample.

## Standard Military Flow Chart (Cont'd.)

| SCREENING | MODIFIED LEVEL S | REQUIREMENT | CLASS B | REQUIREMENT |
|---|---|---|---|---|
| | MIL-STD-883 METHOD 5004 | | MIL-STD-883 METHOD 5004 | |
| Final electrical parameters (hot and cold extremes) | Per applicable device specification | 100% | Per applicable device specification | 100% |
| Sal A) Fine<br>B) Gross | 1014 cond, A or B<br>cond C | 100% | | |
| Group A lot | 5005 Level S. (2) | Per applicable device specification | 5005 Class B (2) | Sample every lot |
| Group B inspection lot<br>Group C<br>Group D<br>Exrternal visual | 5005 level S<br>not applicable<br>5005 level S<br>2009 | As required<br><br>As required<br>100% | 5005 Class B<br>5005 Class B<br>5005 Class B<br>2009<br>Generic data available in lieu of lot quality conformance inspection | Every 6 weeks<br>Every 13 weeks<br>Every 26 weeks<br>100% |

(1) Programming and verification are performed at 25°C only.
(2) Unprogrammable PAL Devices–AC parameters are tested on programmed sample.

## Quality Programs

The Military Product Division quality system conforms to the following Mil-Standards:

Mil-M-38510, Appendix A, "Product Assurance Program"
Mil-Q-9858, "Quality Program Requirements"
Mil-I-45208, "Inspection System Requirements"

Monolithic Memories facilities in Sunnyvale are certified by the Defense Electronics Supply Center (DESC), to manufacture and qualify Schottky Bipolar PROMs and PAL circuits in accordance with Mil-M-38510 Class B. This certification was a result of a successful audit of our production and quality systems to the stringent requirements of Mil-M-38510. Monolithic Memories has also demonstrated compliance to the strict requirements of both controlled and captive lines connected with special Military programs.

## Quality Assurance

Following 100% screening, the Military Products Division samples all products processed in conformance to Mil-Std-883 Class B, to the following LTPD levels:

| Test | LTPD |
|---|---|
| DC 25°C | 2 |
| DC+125°C | 3 |
| DC-55° | 5 |
| Functional at 25°C | 2 |
| Functional at Temperature Extremes | 5 |
| AC 25°C | 2 |
| AC +125°C | 3 |
| AC –55° | 5 |

The Military Products Division ensures outgoing product quality and integrity by performing inspection Lot Group A's and B's per Mil-Std-883 Method 5005, conducting self audits in all areas involved in screening tests per Method 5004 of MIL-STD-883, gating all shipments to our customers, and maintaining a calibration control system in accordance with Mil-Std-45662.

For products requiring programming prior to AC tests, testing is performed utilizing MIL-M-38510 slash sheet sample plans.

## Product Qualification/Quality Conformance Inspection (QCI)

The Military Products Division has a quality conformance testing program in accordance with MIL-STD-883, Method 5005. Quality Conformance Testing provides necessary feedback and monitors several areas:

- Reliability of Product/Processes
- Vendor Qualification for Raw Materials
- Customer Quality Requirements
- Maintain Product Qualification
- Engineering Monitor on Products/Processes

Standard procedures for new product release specify that Monolithic Memories' Reliability Department, as a minimum, conduct qualification testing per Mil-Std-883, Method 5005. Once qualified, each package type (from each assembly line) and device (by technology group as delineated in Mil-M-38510) are incorporated into Monolithic Memories Quality Conformance Inspection program which utilizes the requirements of MIL-M-38510.

When military programs do not require that QCI data be run on the specific lot shipped Monolithic Memories Quality Conformance program allows customers to obtain generic data on all product families manufactured by the Military Products Division. Generic Qualification Data enables customers to eliminate costly qualification and destruct unit charges, and also improves delivery time by a factor of eight to ten weeks. The following product data is available:

### Group B - Package Related Tests

- QCI is performed every 6 weeks of manufacture on each package type.
- Any device type in the same package type may be used regardless of the specific part number.
- Purpose: To monitor assembly and device package integrity.

### Group C - Product/Process Related Tests

- QCI is performed every 13 weeks of manufacture, on representative devices from the same microcircuit group.
- Life test data may be used to qualify similar technologies.
- Purpose: To monitor the reliability of the process and the parametric performance for each product technology.

### Group D - In-Depth Package Related Tests

- QCI is conducted every 26 weeks using devices which represent the same package construction and lead finish.
- Any device type in the same package type may be used regardless of the specific part number.
- Purpose: To monitor the reliability and integrity of various package materials and assembly processes.

### Generic Data

Monolithic Memories' Generic Data Program is based on MIL-M-38510, which allows for shipments based on 26 weeks of coverage for Group C Testing and 36 weeks of coverage for Group D Testing.

Should circumstances arise where generic coverage to MIL-M-38510 is not possible, MMI reserves the right to ship product based on 52 weeks of generic Group C and/or D coverage per MIL-Std-883.

## Process Audits

Process Audits are performed in accordance with Mil-M-38510, Appendix A, (self audits) by the Quality Assurance Department.

## Customer Material Returns

In order to better service our military customers who must return product to the factory, the Military Products Division has established its own customer material returns department. Our goals and policies are outlined below so you may know what to expect when returning product to M.P.D.

### Goals

10 day turn-around to respond to a return.

- Notification to the customer of any discrepancy relating to the return.
- For returns which cannot be validated, a written notice of M.P.D.'s intent to return product will be sent to our customer.
- Product returned to our customer will be accompanied by an explanation and/or parametric test data and serialized devices.

### Standard Policies

- Product which is returned specifically as electrical failures and is not accompanied by test data, will be tested at all three temperatures ( –55°C, 25°C, +125°C)
- If no failures found, the product will be returned to the customer.
- A device count is done upon arrival of a return at Monolithic Memories. Credit will be given only for the number of devices received by the factory.
- Product returned by the Franchised Distributor for rescreen or stock rotation will be accepted only if proper traceability paperwork accompanies each lot of product
- All returns must be send to 1135 East Arques, Sunnyvale, Ca 94086. ATTN: "MPD CMR DEPARTMENT"

### Information Checklist

The following accompanying a material return will assist us in responding to your return in the shortest possible time.

1. Double check accuracy of device counts.
2. Identify rejects from good devices, when returned together.
3. Supply as much detail as you can about the description of the electrical failure mode (i.e.: AC Fail, DC Fail, FCT Fail, or description of any test numbers used).
4. Whenever possible, identify dissimilar failures or keep separate devices which fail different parameters by serializing failures.
5. Enclose a copy of any data which you may have taken on the failed devices (i.e.: Forcing conditions, temperature tested, parameter and value, an address that failed). List what was expected vs. what was received.
6. For programmability failures, please send programming masters or a truth table. Also please indicate whether single or multiple pulse programming was used and the equipment device was programmed on.
7. What environmental testing was performed.
8. Failure rates.

Although our intent is that our customers will never have to use these guidelines, if a problem should arise, the Military Products Division will strive to disposition and respond to your material return as thoroughly and promptly as possible.

## Electrostatic Discharge Control Procedures

The Military Products Division of Monolithic Memories fully employs static control procedures throughout its facilities in Penang, Malaysia and Sunnyvale, California.

All manufacturing areas where product is processed or handled, including our Reliability Labs, Engineering Labs, etc., have full static control such as wrist straps, antistatic smocks, grounded stainless steel tables, conductive mats and ion generators wherever necessary.

All product is moved throughout our facilities and shipped to customers in static shielded containers.

In addition, MPD distributors must demonstrate that they meet the same stringent standards regarding ESD handling and control procedures as the factory. Individual distributor locations are audited and approved annually by MPD's Quality Assurance Department.

An ESD identifier is marked on all products in front of the date code, and all shipping containers are labeled with an ESD Caution Message. ESD procedures are continually reviewed, to ensure that our customers receive only the highest quality product from the Military Products Division.

## Radiation Hardness Program

### 1. Radiation Effects

It has been stated that some level of radiation tolerance will be required in up to 50% of all military applications by 1990. Due to this increased concern over radiation effects on integrated circuits, the Military Products Division has embarked on a program to determine what radiation dose rates our circuits will withstand.

### 2. Neutron Irradiation

We have successfully completed neutron radiation testing on our Bipolar processes in accordance with Mil-Std-883, Method 1017.2. Eleven different device types, which currently represent all our Bipolar processes, were parametrically and functionally tested at 25°C before and after exposure to fluence levels of $2x10^{12}$ N/cm$^2$, $1x10^{13}$ N/cm$^2$, $4x10^{13}$ N/cm$^2$ and $1x10^{14}$ N/cm$^2$. Input low current (IIL) is the primary measurement of permanent circuit degradation. The parametric failures (IIL > 250mA) seen occurred at relatively high fluence levels. Also, no major changes in ICC were noted for any circuit.

The following is a list of the device types tested:

| | |
|---|---|
| 53S1681 | (2048x8 PROM) |
| 53RA1681A | (2048x8 Registered PROM) |
| 53S3281 | (4096x8 PROM) |
| 57401A | (64x4 FIFO) |
| PAL16R4A | (High Speed Programmable Array Logic) |
| PAL16R4B | (Very High Speed Programmable Array Logic) |
| PAL16R4D | (Oxide Isolated Ultra High Speed Programmable Array Logic) |
| PAL20R4A | (High Speed Programmable Array Logic) |
| PAL20RA10 | (Asynchronous Programmable Array Logic) |

All devices passed test limits at $1x10^{13}$ N/cm² level, with the 53RA1681A, 53S3281, and 57401A also passing the $4x10^{13}$ N/cm² fluence level. In addition, the 57401A passed test limits at $1x10^{14}$ N/cm² level.

### *3.* Dose Rate Effects

Dose rate data has been obtained on our junction isolated Bipolar processes. All recovered in 50 to 70 microseconds from a 1 microsecond pulse of $2x10^{10}$ rads (Si) per second.

The products tested were:

| | |
|---|---|
| PAL14L8 | PAL10L8 |
| PAL16L6 | PAL12L6 |
| PAL20L10 | PAL16L8 |
| PAL20X10 | PAL16R8 |
| PAL20X8 | PAL16R6 |
| PAL12H6 | PAL16R4 |
| PAL14H4 | |

### 4. Future Radiation Testing

Our future test plans include:

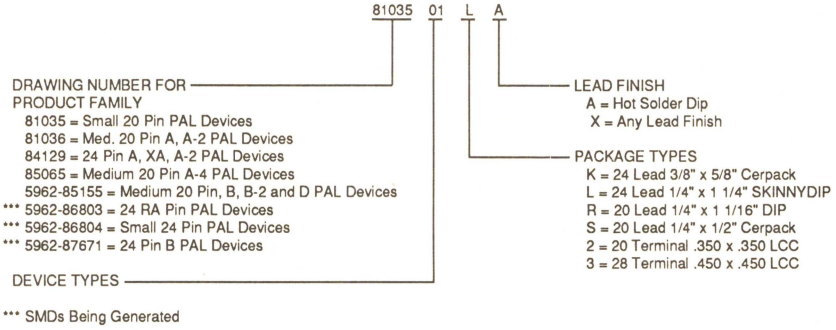
- Total Dose
- Single Event Upset
- Latch-up and Burn-out

Monolithic Memories' new Bipolar and CMOS processes will be radiation tested after production release.

Detailed neutron and dose rate radiation data is available from the Military Products Division.

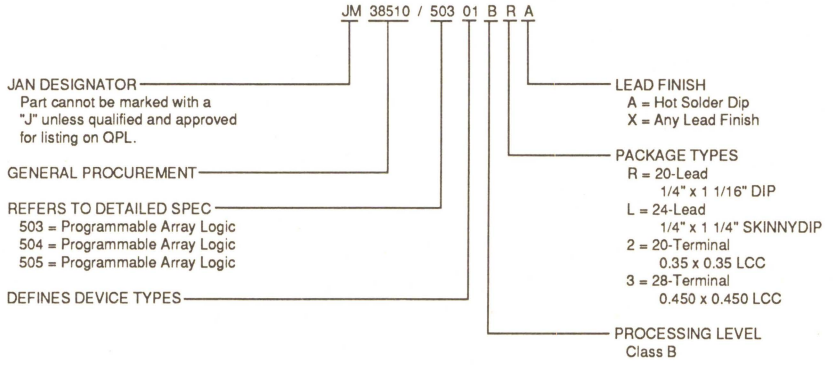
# JAN 38510 and STANDARD MILITARY DRAWING PROGRAM

## STANDARD MILITARY DRAWING NUMBERING SYSTEM



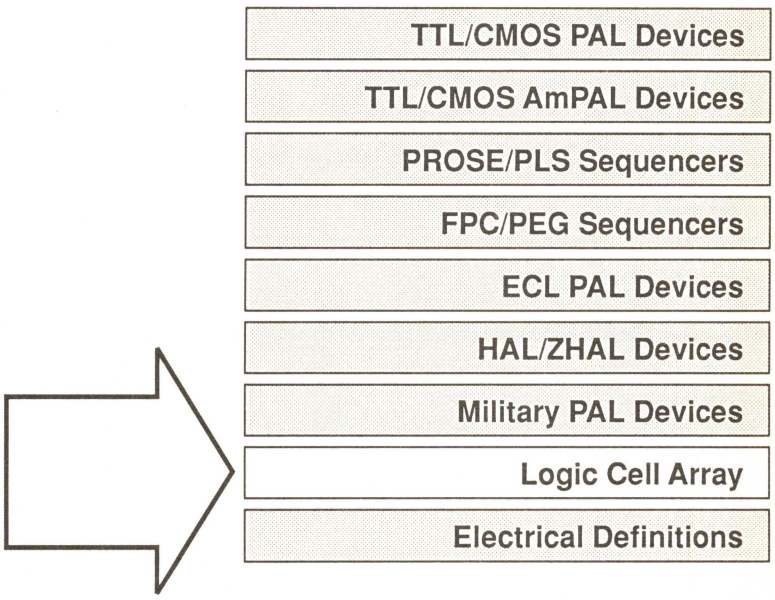

*** SMDs Being Generated

## JAN PART NUMBERING SYSTEM




PART NUMBER INTERPRETATION:
When ordering to JAN 38510 and Military Drawing numbers, the lead finish designator (last letter in part number) is commonly called out as "X". This is a way of stating that the customer will accept the standard manufacturer's lead finish for the package orders. "X" is not a lead finish designator in itself, therefore, when product is shipped, the actual lead finish designator will be marked on the devices.

## Data Sheets




5


IMPORTANT
NOTES

# Logic Cell™ Array M2064/M2018

## Features/Benefits

- CMOS programmable Logic Cell Array (LCA) for replacement of standard logic
- Completely reconfigurable by the user in the final system
- High performance equivalent to TTL SSI/MSI
  - 33 MHz flip-flop toggle rate (-33 speed grade)
  - 50 MHz flip-flop toggle rate (-50 speed grade)
  - 70 MHz flip-flop toggle rate (-70 speed grade)
- User configurable logic functions, interconnect and I/O for maximum flexibility
- 64/100 user-Configurable Logic Blocks (CLBs) providing usable gate equivalency of up to 1200/1800 gates
- 58/74 individually-configurable I/O pins allowing any mix of inputs, outputs or bidirectional signals (68/84-pin package)
- User -selectable TTL or HCMOS input threshold levels
- Multiple configuration modes for greatest flexibility and ease-of-use
- Verification feature allows user to check configuration data
- User-selectable security feature prevents read-back of configuration data
- Read-back of internal register states for system debug
- On-chip clock oscillator and clock buffer circuits provide flexible internal and external clocking functions
- Master reset of all internal register elements in addition to user-configurable Reset and/or Set control of individual CLB storage elements
- Complete development system support

## General Description

The Logic Cell Array (LCA) is a high-density CMOS-integrated circuit available from Monolithic Memories. Its user-programmable array architecture is made up of three types of configurable elements: Input/Output Blocks, Logic Blocks and Interconnect. The designer can define individual I/O blocks for interface to external circuitry, define logic blocks to implement logic functions and define interconnection network to compose larger scale logic functions. The XACT™ Development system provides interactive graphic design capture and automatic routing. Both logic simulation and in-circuit emulation are available for design verification.

The Logic Cell Array is available in a variety of logic capacities, package styles, temperature ranges and speed grades.

| PART NUMBER | LOGIC CAPACITY (USABLE GATES) | CONFIG-URABLE LOGIC BLOCKS | USER I/Os | CONFIG-URATION PROGRAM (BITS) |
|---|---|---|---|---|
| M2064 | 1200 | 64 | 58 | 12040 |
| M2018 | 1800 | 100 | 74 | 17880 |

The Logic Cell Array's logic functions and interconnections are determined by data stored in internal static memory cells. On-chip logic provides for automatic loading of configuration data at power-up. The program data can reside in an EEPROM, EPROM or ROM on the circuit board or on a floppy disk or hard disk. The program can be loaded in a number of modes to accommodate various system requirements.

## Package Availability

| PART NUMBER | 48-PIN PLASTIC DIP N48 | 68-PIN PLCC NL68 | 68-PIN PGA P68 | 84-PIN PLCC NL84 | 84-PIN PGA P84 |
|---|---|---|---|---|---|
| M2064 | X | X | X | | |
| M2018 | | X | X | X | X |

## Ordering Information

M2018-70 C NL 84

- PART NUMBER
  - 2064 (1200 Gates, 58 IOB)
  - 2018 (1800 Gates, 74 IOB)
- SPEED GRADE
  - -33 (33 MHz Toggle Rate)
  - -50 (50 MHz Toggle Rate)
  - -70 (70 MHz Toggle Rate)
- NUMBER OF PINS
  - 48 (48 Pins)
  - 68 (68 Pins)
  - 84 (84 Pins)
- PACKAGE TYPE
  - NL = Pin Molded Chip Carrier
  - P = Pin Grid Array
  - N = Pin Molded DIP
- TEMPERATURE RANGE
  - C = Commercial
  - M = Military

XILINX™, XACT™, XACTOR™, Logic Cell™ Array and Logic Processor™ are trademarks of XILINX Inc.

IBM® is a registered trademark of International Business Machines Corporation.

PC™, PC-AT™ and PC-XT™ are trademarks of International Business Machines Corporation.

P-SILOS™ is a trademark of SimuCad Corporation.

MS-DOS™ is a trademark of Microsoft Corporation.

Portions of this Data Sheet reproduced with the permission of XILINX Inc.

10352A
JANUARY 1988



5

## Pin Description

**$\overline{\text{PWRDWN}}$**
An active low power-down input stops all internal activity to minimize VCC power and puts all output buffers in a high-impedance state. Configuration is retained, however, internal storage elements are Reset. When the $\overline{\text{PWRDWN}}$ pin returns HIGH, the device returns to operation with the same sequence of reset, buffer enable and DONE, $\overline{\text{PROGRAM}}$ as at the completion of configuration.

**M0, RTRIG**
As Mode 0, this input and M1, M2 are sampled before the start of configuration to establish the configuration mode to be used.

As a read trigger, an input transition to a HIGH, after configuration is complete, will initiate a readback of configuration and storage element data. This operation may be limited to a single request, or be inhibited altogether, by selecting the appropriate readback option when generating the bit stream.

**M1, $\overline{\text{RDATA}}$**
As Mode 1, this input and M0, M2 are sampled before the start of configuration to establish the configuration mode to be used.

As an active-low read data; after configuration is complete, this pin is the output of the readback data.

**M2**
As Mode 2, this input and M0, M1 are sampled before the start of configuration to establish the configuration, mode to be used. After configuration, this pin becomes a user-programmable I/O.

**HDC**
High during configuration is held at a HIGH level by the LCA until after configuration. It is intended to be available as a control indication that configuration is not complete. After configuration, this pin is a user I/O.

**$\overline{\text{LDC}}$**
Low during configuration is held at a LOW level by the LCA until after configuration. It is intended to be available as a control indication that configuration is not completed. It is particularly useful in master mode as a LOW enable for an EPROM. After configuration, this pin is a user I/O. If used as a LOW EPROM enable, it should be programmed as a HIGH after configuration.

**$\overline{\text{RESET}}$**
This is an active-low input which has three functions. Prior to the start of configuration, a LOW input will delay the start of the configuration process. An internal circuit senses the application of power and begins a minimal time-out cycle on the order of 100 ms. When the time-out and $\overline{\text{RESET}}$ are complete, the levels of the "M" mode lines are sampled and configuration begins. If $\overline{\text{RESET}}$ is asserted during a configuration, the LCA is reinitialized and will restart the configuration at the termination of $\overline{\text{RESET}}$. If $\overline{\text{RESET}}$ is asserted after configuration is complete, it will provide an asynchronous reset of all IOB and CLB storage elements of the LCA.

**DONE, $\overline{\text{PROG}}$**
The DONE open drain output is configurable with or without a pull-up resistor of about 3 KΩ. At the completion of configuration, the circuitry of the LCA becomes active in a synchronous order and one configuration clock cycle later DONE is asserted. Once configuration is done, a HIGH-to-LOW transition of this program pin will cause an initialization of the LCA and start a reconfiguration if that mode is selected in the current configuration.

**XTL1**
This user I/O pin may be configured to operate as the output of an amplifier usable with an external crystal and bias circuitry to form an oscillator.

**XTL2**
This user I/O pin may be configured to operate as the input of an amplifier usable with an external crystal and bias circuitry to form an oscillator.

**CCLK**
During configuration, configuration clock is an output of an LCA in either master or peripheral mode. LCAs in slave mode use it as a clock input. During a readback operation, it is an input clock for the configuration data being output.

**DOUT**
This user I/O pin is used during configuration to output serial configuration data out for daisy-chained slaves' data in.

**DIN**
This user I/O pin is used as serial data in during slave or peripheral configuration. This pin is D0 in master configuration mode.

**$\overline{\text{CSO}}$, $\overline{\text{CS1}}$, CS2, $\overline{\text{WRT}}$**
These four inputs represent a set of signals, three active low and one active high, which are used in the peripheral mode to control configuration data entry. The assertion of all four generates a LOW CCLK and shifts DOUT data. The removal of any assertion clocks in the DIN data present and causes a HIGH CCLK. In master mode, these pins become part of the parallel configuration byte (D4, D3, D2, D1). After configuration is complete, they are user-programmed I/O.

**$\overline{\text{RCLK}}$**
During master mode configuration, this pin represents a read clock of an external memory device. After configuration is complete, this pin becomes a user-programmed I/O.

**D0-D7**
This set of eight pins represents the parallel configuration data byte for the master mode. After configuration is complete, they are user-programmed I/O.

**A0-A15**
This set of sixteen pins presents an address output for an external configuration memory during master mode. After configuration is complete, they are user-programmed I/O. A12 through A15 are not available in packages with less than sixty-eight pins.

**I/O**
A pin which may be programmed by the user to be input and/or output following configuration. Some of these pins present a high-impedance pull-up or perform other functions before configuration is complete.

## Functional Description

The general structure of a Logic Cell Array is shown in Figure 1. The elements of the array include three categories of user-programmable elements: I/O Blocks, Configurable Logic Blocks and Programmable Interconnections. The I/O Blocks provide an interface between the logic array and the device package pins. The Configurable Logic Blocks perform user-specified logic functions, and the interconnect resources are programmed to form networks that carry logic signals among blocks.

Configuration of the Logic Cell Array is established through a distributed array of memory cells. The XACT development system generates the program used to configure the Logic Cell Array. The Logic Cell Array includes logic to implement automatic configuration.

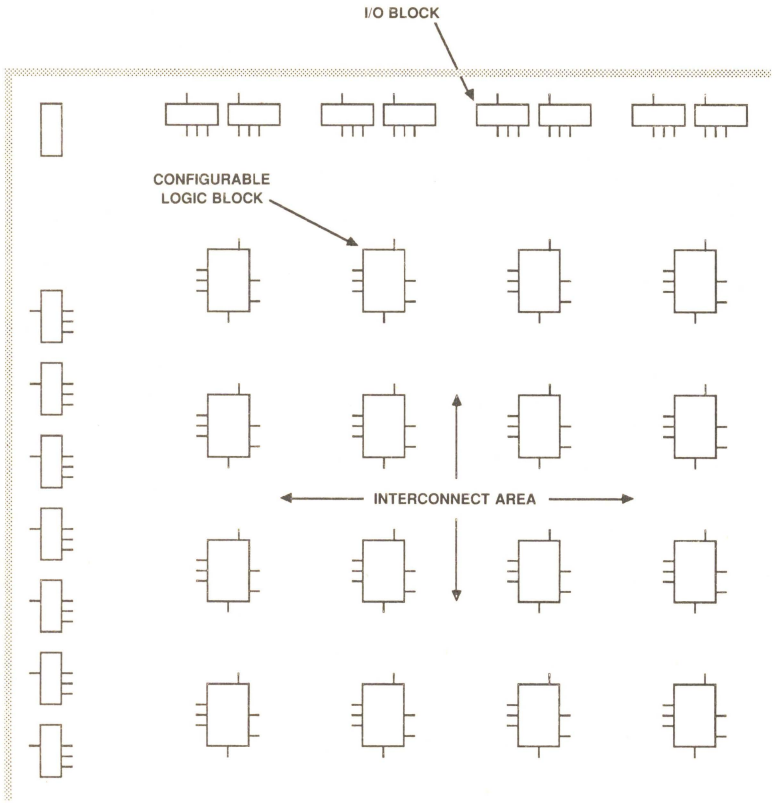



Figure 1. Logic Cell Array Structure

### Configuration Memory

The configuration of the Monolithic Memories' Logic Cell Array is established by programming memory cells which determine the logic functions and interconnections. The memory loading process is independent of the user logic functions.

The static memory cell used for the configuration memory in the Logic Cell Array has been designed specifically for high reliability and noise immunity. Based on this design, integrity of the LCA configuration memory is assured even under adverse conditions. Compared with other programming alternatives, static memory provides the best combination of high density, high performance, high reliability and comprehensive testability. As shown in Figure 2, the basic memory cell consists of two CMOS inverters plus a pass transistor used for writing data to the cell. The cell is only written during configuration and only read during readback. During normal operation the pass transistor is "off" and does not affect the stability of the cell. This is quite different from the normal operation of conventional memory devices, in which the cells are continuously read and written.

The outputs Q and $\overline{Q}$ control pass-transistor gates directly. The absence of sense amplifiers and the output capacitive load provide additional stability to the cell. Due to the structure of the configuration memory cells, they are not affected by extreme power supply excursions or very high levels of alpha particle radiation. In reliability testing no soft errors have been observed, even in the presence of very high doses of alpha radiation.

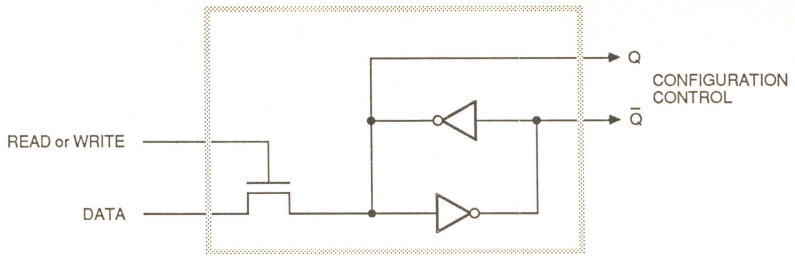



**Figure 2. Configuration Memory Cell**

### Input/Output Block

Each user-configurable I/O block (IOB) provides an interface between the external package pin of the device and the internal logic. Each I/O block includes a programmable input path and a programmable output buffer. It also provides input clamping diodes to provide protection from electrostatic damage, and circuits to protect the LCA from latch-up due to input currents. Figure 3 shows the general structure of the I/O block.

The input buffer portion of each I/O block provides threshold detection to translate external signals applied to the package pin to internal logic levels. The input buffer threshold of the I/O blocks can be programmed to be compatible with either TTL (1.4 V) or CMOS (2.2 V) levels. The buffered input signal drives both the data input of an edge-triggered D-type flip-flop and one input of a two-input multiplexer. The output of the flip-flop provides the other input to the multiplexer. The user can select either the direct input path or the registered input, based on the content of the memory cell controlling the multiplexer. The I/O blocks along each edge of the die share common clocks. The flip-flops are reset during configuration as well as by the active-low chip $\overline{\text{RESET}}$ input.

Output buffers in the I/O blocks provide 4-mA drive for high fan-out CMOS or TTL-compatible signal levels. The output data (driving I/O block pin O) is the data source for the I/O block output buffer. Each I/O block output buffer is controlled by the contents of two configuration memory cells which turn the buffer ON or OFF or select logical three-state buffer control. The user may also select the output buffer three-state control (I/O block pin TS). When this I/O block output control signal is HIGH (a logic "1") the buffer is disabled and the package pin is high-impedance.

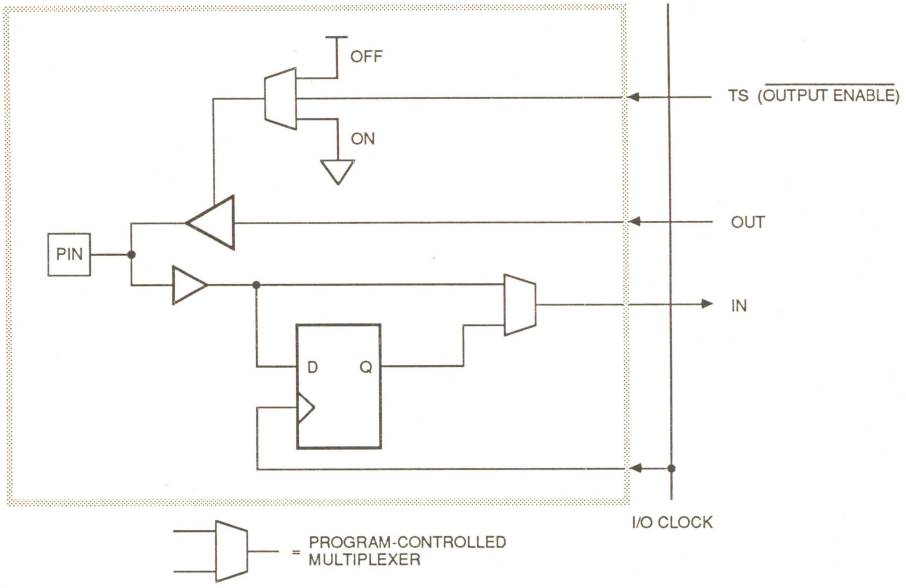



**Figure 3. I/O Block**

### Configurable Logic Block

An array of Configurable Logic Blocks (CLBs) provides the functional elements from which the user's logic is constructed. The logic blocks are arranged in a matrix in the center of the device. The M2064 has sixty-four such blocks arranged in an eight-row by eight-column matrix. The M2018 has one hundred logic blocks arranged in a ten by ten matrix. Each logic block has a combinatorial logic section, a storage element, and an internal routing and control section. Each CLB has four general-purpose inputs; A, B, C and D; and a special clock input (K), which may be driven from the interconnect adjacent to the block. Each CLB also has two outputs, X and Y, which may drive interconnect networks. Figure 4 shows the resources of a Configurable Logic Block.

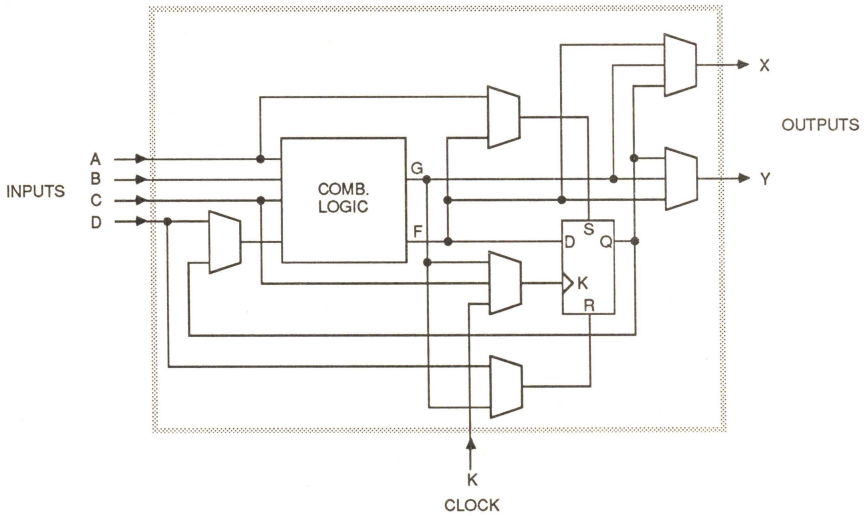



**Figure 4. Configurable Logic Block**

The logic block combinatorial logic uses a table look-up memory to implement Boolean functions. This technique can generate any logic function of up to four variables with a high-speed sixteen-bit memory. The propagation delay through the combinatorial network is independent of the function generated. Each block can perform any function of four variables or any two functions of three variables each. The variables may be selected from among the four inputs and the block's storage element output "Q". Figure 5 shows various options which may be specified for the combinatorial logic.

5

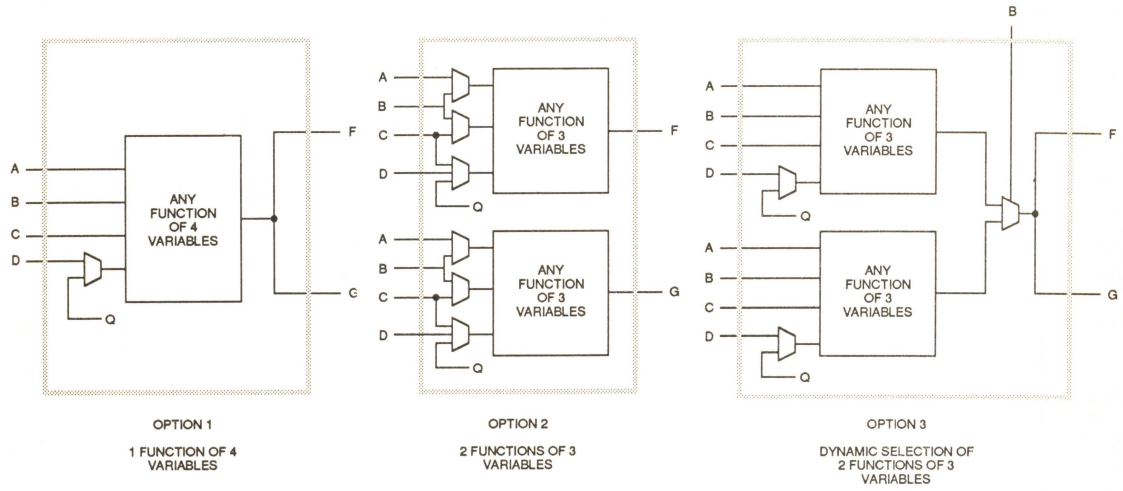



**Figure 5. CLB Combinatorial Logic Options**

If the single four-variable configuration is selected (Option 1), the F and G outputs are identical. If the two-function alternative is selected (Option 2), logic functions F and G may be independent functions of three variables each. The three variables can be selected from among the four logic block inputs and its storage element output Q. A third form of the combinatorial logic (Option 3) is a special case of the two-function form in which the B input dynamically selects between the two function tables providing a single merged logic function output. This dynamic selection allows some five-variable functions to be generated from the four block inputs and storage element Q. Combinatorial functions are restricted in that one may not use both its storage element output Q and the input variable of the logic block pin D in the same function.

If used, the storage element in each Configurable Logic Block (Figure 6) can be programmed to be either an edge-sensitive D-type flip-flop or a level-sensitive D latch. The clock or enable for each storage element can be selected from:

- The special-purpose clock input K
- The general-purpose input C
- The combinatorial function G

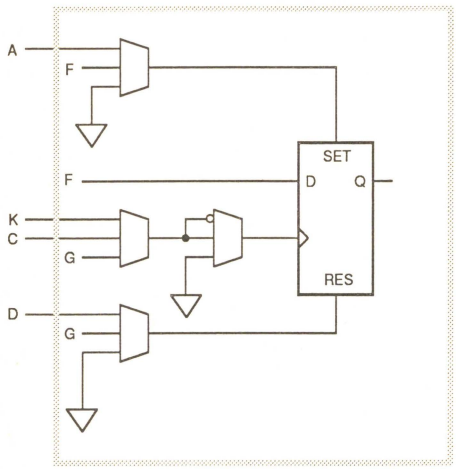



**Figure 6. CLB Storage Element**

The user may also select the clock active sense within each logic block. This programmable inversion eliminates the need to route both phases of a clock signal throughout the device.

The storage element data input is supplied from the function F output of the combinatorial logic. Asynchronous SET and RESET controls are provided for each storage element. The user may enable these controls independently and select their source. They are active-high inputs and the asynchronous reset is dominant. The storage elements are reset by the active-low chip $\overline{\text{RESET}}$ pin as well as by the initialization phase preceding configuration. If the storage element is not used, it is disabled.

The two block outputs, X and Y, can be driven by either the combinatorial functions, F or G, or the storage element output Q (Figure 4). Selection of the outputs is completely interchangeable and may be made to optimize routing efficiencies of the networks interconnecting the logic blocks and I/O blocks.

**Programmable Interconnect**

Programmable interconection resources in the Logic Cell Array provide routing paths to connect inputs and outputs of the I/O and logic blocks into desired networks. All interconnections are composed of metal segments, with programmable switching points provided to implement the necessary routing. Three types of resources accommodate different types of networks:

- General purpose interconnect
- Long lines
- Direct connection

**General-Purpose Interconnect**

General-purpose interconnect, as shown in Figure 7a, is composed of four horizontal metal segments between the rows and five vertical metal segments between the columns of logic and I/O blocks. Each segment is only the "height" or "width" of a logic block. Where these segments would cross at the intersections of rows and columns, switching matrices are provided to allow interconnections of metal segments from the adjoining rows and columns. Switches in the switch matrices and on block outputs are specially designed transistors, each controlled by a configuration bit.

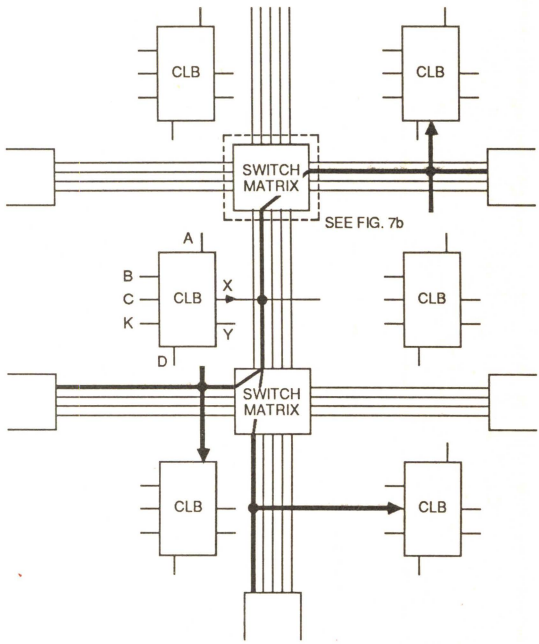
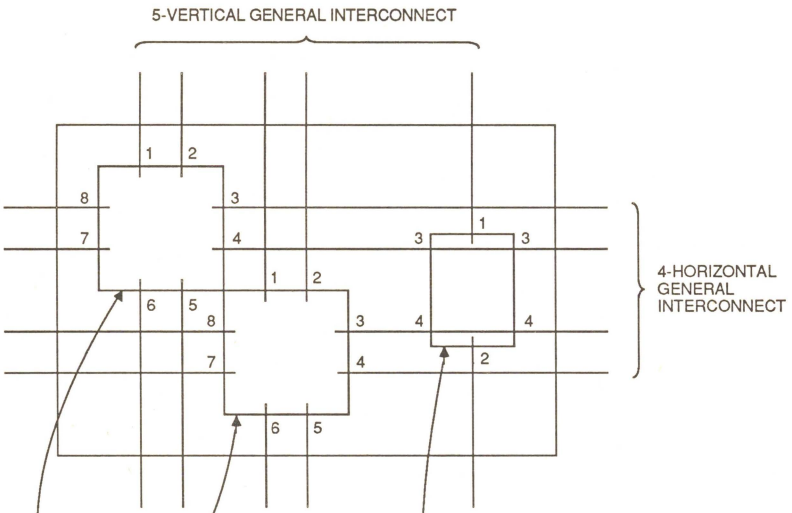



**Figure 7a. General-Purpose Interconnect**

Logic block output switches provide contacts to adjacent general interconnect segments and therefore to the switching matrix at each end of those segments. A switch matrix can connect an interconnect segment to other segments to form a network. Figure 7a shows the general interconnect used to route a signal from one logic block to three other logic blocks. As shown, combinations of closed switches in a switch matrix allow multiple branches for each network. The inputs of the logic or I/O blocks are multiplexers that can be programmed with configuration bits to select an input network from the adjacent interconnect segments. Since the switch connections to block inputs are undirectional (as are block outputs) they are usable *only* for input connections. The development system software provides automatic routing of these interconnections. Interactive routing is also available for design optimization. This is accomplished by selecting a network and then toggling the states of the interconnect points by selecting them with the "mouse". In this mode, the connections through the switch matrix may be established by selecting pairs of matrix pins. The switching matrix combinations are indicated in Figure 7b.

Special buffers within the interconnect area provide periodic signal isolation and restoration for higher general interconnect fan-out and better performance. The repowering buffers are bidirectional, since signals must be able to propagate in either direction on a general interconnect segment. Direction controls are automatically established by the Logic Cell Array development system software. Repowering buffers are provided only for the general-purpose interconnect since the direct and long-line resources do not exhibit the same R-C delay accumulation. The Logic Cell Array is divided into nine sections with buffers automatically provided for general interconnect at the boundaries of these sections. These boundaries can be viewed with the development system. For routing within a section, no buffers are used. The delay calculator of the XACT development system automatically calculates and displays the block, interconnect and buffer delays for any selected paths.




FROM / TO

| TO \ FROM | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | ■ | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | ■ | 1 | 1 | 1 | 1 | 0 | 1 |
| 3 | 1 | 1 | ■ | 0 | 1 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | ■ | 1 | 1 | 1 | 1 |
| 5 | 1 | 1 | 1 | 1 | ■ | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 1 | 0 | ■ | 1 | 1 |
| 7 | 1 | 0 | 1 | 1 | 1 | 1 | ■ | 0 |
| 8 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | ■ |

FROM / TO

| TO \ FROM | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | ■ | 1 | 1 | 1 |
| 2 | 1 | ■ | 1 | 1 |
| 3 | 1 | 1 | ■ | 0 |
| 4 | 1 | 1 | 0 | ■ |

1 = VALID CONNECTION
0 = INVALID CONNECTION

**Figure 7b. Interconnection Switching Matrix**

### Long Lines

Long lines, shown in Figure 8a, run both vertically and horizontally the height or width of the interconnect area. Each vertical interconnection column has two long lines; each horizontal row has one, with an additional long line adjacent to each set of I/O blocks. The long lines bypass the switch matrices and are intended primarily for signals that must travel a long distance or must have minimum skew among multiple destinations.

A global buffer in the Logic Cell Array is available to drive a single signal to all B and K inputs of logic blocks. Using the global buffer for a clock provides a very low skew, high fan-out synchronized clock for use at any or all of the logic blocks. At each block, a configuration bit for the K input to the block can select this global line as the storage element clock signal. Alternatively, other clock sources can be used.

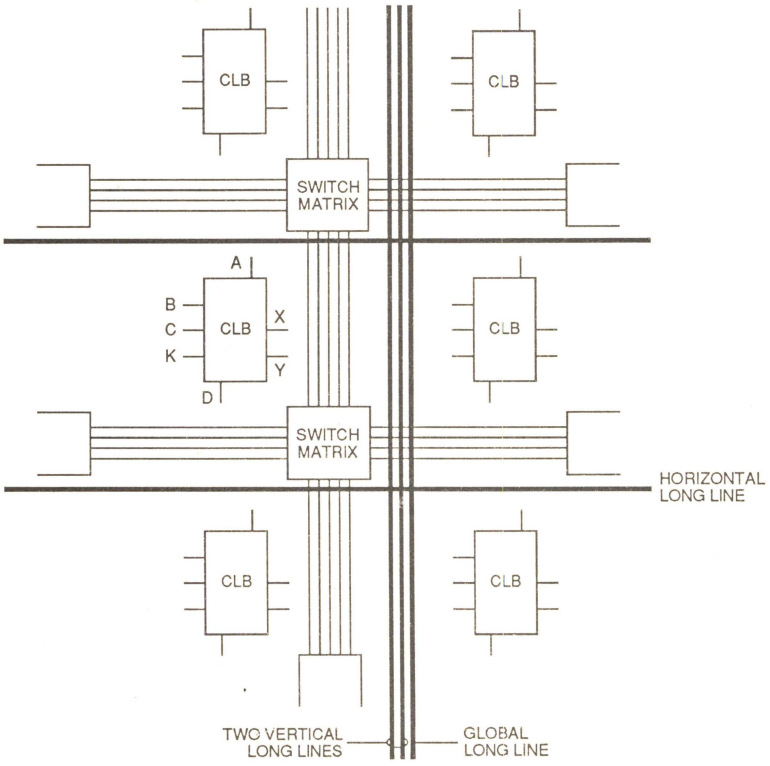



**Figure 8a. Long Line Interconnect**

A second buffer below the bottom row of the array drives a horizontal long line which, in turn, can drive a vertical long line in each interconnection column. This alternate buffer also has low skew and high fan-out capability. The network formed by this alternate buffer's long lines can be selected to drive the B, C or K inputs of the logic blocks. Alternatively, these long lines can be driven by a logic or I/O block on a column-by-column basis. This capability provides a common, low-skew clock or control line within each column of logic blocks. Interconnections of these long lines are shown in Figure 8b.

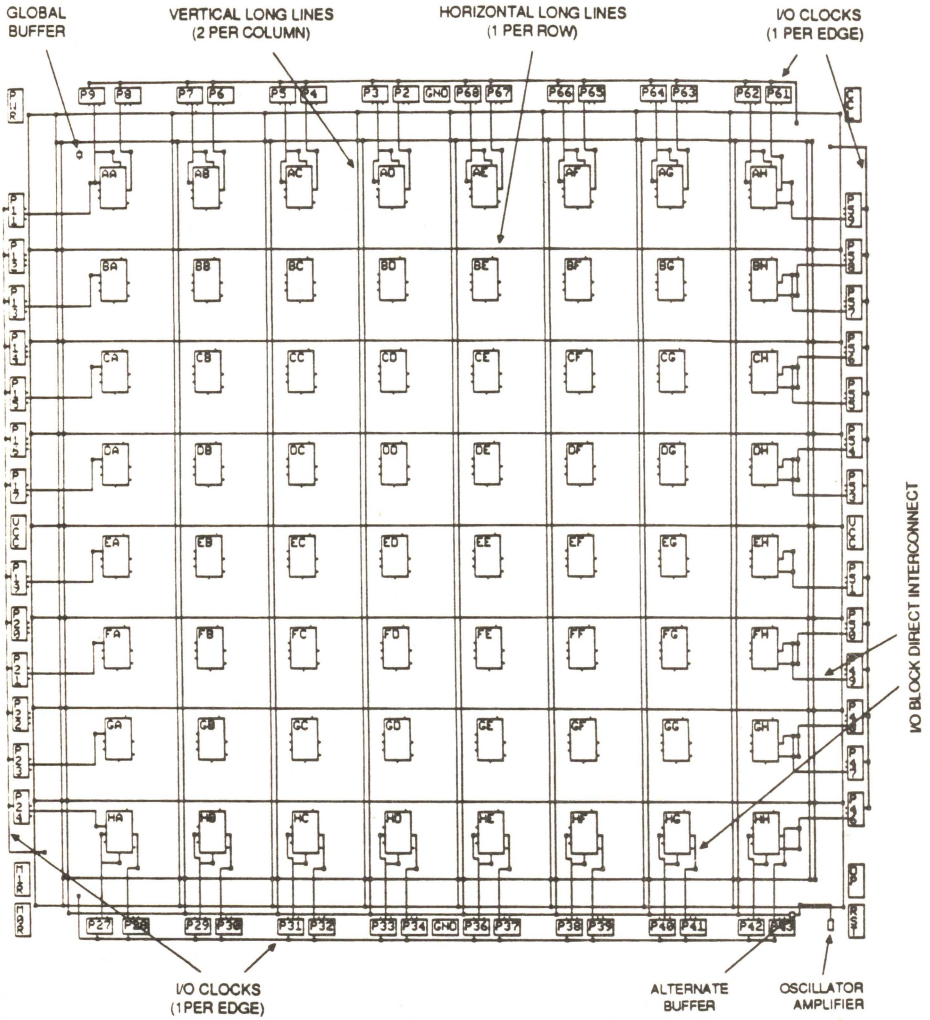



Figure 8b. M2064 Long Lines, I/O Clocks, I/O Direct Interconnect

### Direct Interconnect

Direct interconnect, shown in Figure 9, provides the most efficient implementation of networks between adjacent logic or I/O blocks. Signals routed from block to block by means of direct interconnect exhibit minimum interconnect propagation and use minimum interconnect resources. For each CLB, the X output may be connected directly to the C or D inputs of the CLB above and to the A or B inputs of the CLB below it. The Y output can use direct interconnect to drive the B input of the block immediately to its right. Where logic blocks are adjacent to I/O blocks, direct connect is provided to the I/O block input (I) on the left edge of the die, the output (O) on the right edge, or both on I/O blocks at the top and bottom of the die. Direct interconnections of I/O blocks with CLBs are shown in Figure 8b.

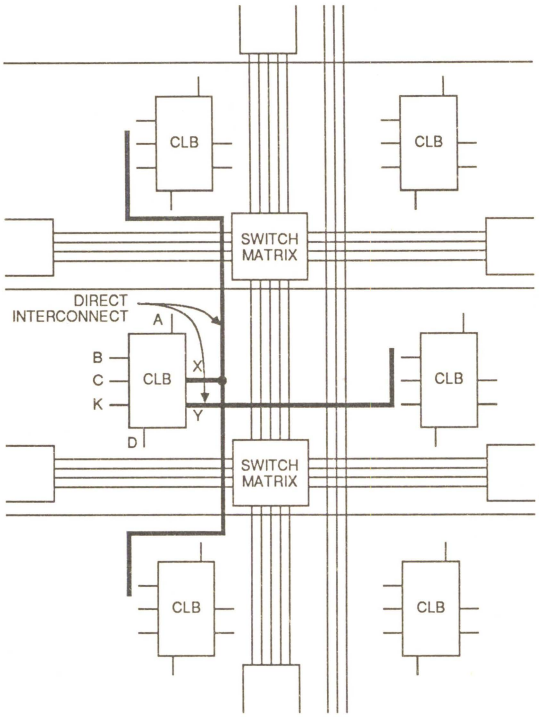
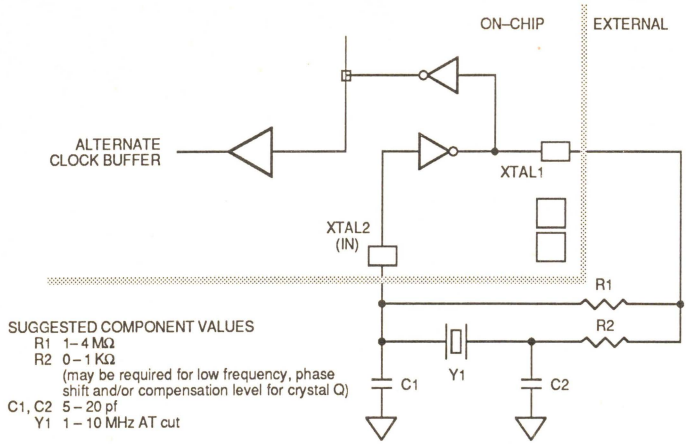



Figure 9. Direct Interconnect

### Crystal Oscillator

An internal high-speed inverting amplifier is available to implement an on-chip crystal oscillator. It is associated with the auxiliary clock buffer in the lower right corner of the die. When configured to drive the auxiliary clock buffer, two special adjacent user I/O blocks are also configured to connect the oscillator amplifier with external crystal oscillator components, as shown in Figure 10. This circuit becomes active before configuration is complete in order to allow the oscillator to stabilize. Actual internal connection is delayed until completion of configuration. The feedback resistor R1 between output and input, biases the amplifier at threshold. It should be as large a value as practical to minimize loading of the crystal. The inversion of the amplifier, together with the R-C networks and crystal, produces the 360-degree phase shift of the Pierce oscillator.

A series resistor R2 may be included to add to the amplifier output impedance when needed for phase-shift control or crystal resistance matching or to limit the amplifier input swing to control clipping at large amplitudes. Excess feedback voltage may be adjusted by the ratio of C2/C1. The amplifier is designed to be used over the range from 1 MHz up to one-half the specified CLB toggle frequency. Use at frequencies below 1 MHz may require individual characterization with respect to a series resistance. Operation at frequencies above 20 MHz generally requires a crystal to operate in a third overtone mode, in which the fundamental frequency must be suppressed by the R-C networks. When the amplifier does not drive the auxiliary buffer, these I/O blocks and their package pins are available for general user I/O.




| | XTAL1 | XTAL2 |
|---|---|---|
| 48 DIP | 33 | 30 |
| 68 PLCC | 46 | 43 |
| 68 PGA | J10 | L10 |
| 84 PLCC | 56 | 53 |
| 84 PGA | K11 | L11 |

**Figure 10. Crystal Oscillator**

## Power

### Power Distribution

Power for the LCA is distributed through a grid to achieve high noise immunity and isolation between logic and I/O. For packages having more than fourty-eight pins, two VCC pins and two ground pins are provided (see Figure 11). Inside the LCA, a dedicated VCC and ground ring surrounding the logic array provides power to the I/O drivers. An independent matrix of VCC and ground lines supplies the interior logic of the device. This power distribution grid provides a stable supply and ground for all internal logic, providing the external package power pins are appropriately decoupled. Typically a 0.1-μF capacitor connected between the VCC and ground pins near the package will provide adequate decoupling.

Output buffers capable of driving the specified 4-mA loads under worst-case conditions may be capable of driving 25 to 30 times that current in a best case. Noise can be reduced by minimizing external load capacitance and reducing simultaneous output transitions in the same direction. It may also be beneficial to locate heavily-loaded output buffers near the ground pads. Multiple VCC and ground pin connections are required for package types which provide them.

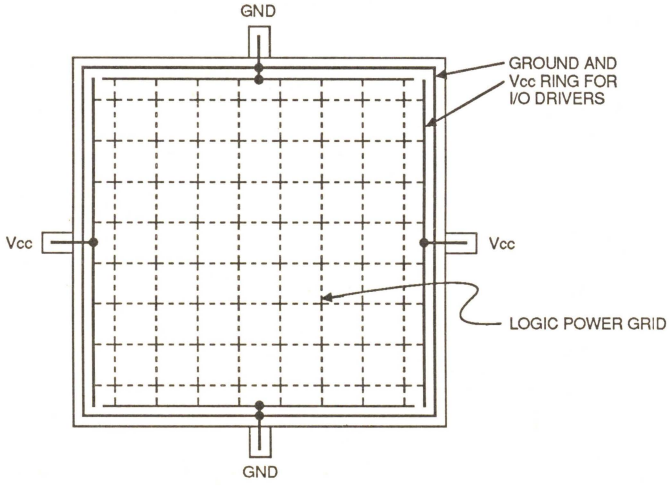



**Figure 11. LCA Power Distribution**

### Power Dissipation

The Logic Cell Array exhibits the low power consumption characteristic of CMOS ICs. Only quiescent power is required for the LCA configured for CMOS input levels. The TTL input level configuration option requires additional power for level shifting. The power required by the static memory cells which hold the configuration data is very low and may be maintained in a power-down mode.

Typically most of power dissipation is produced by capacitive loads on the output buffers, since the power per output is 25 µW/pF/MHz. Another component of I/O power is the DC loading on each output pin. For any given system, the user can calculate the power requirement based on the resistive loading of the devices driven by the Logic Cell Array.

Internal power supply dissipation is a function of clock frequency and the number of nodes changing on each clock. In an LCA the fraction of nodes changing on a given clock is typically low (10-20%). For example, in a 16-bit binary counter, the average clock produces a change in slightly less than two of the sixteen bits. In a 4-input AND gate there will be two transitions in sixteen states. Typical global clock buffer power is about 3 mW/MHz for the M2064 and 4 mW/MHz for the M2018. With a "typical" load of three general interconnect segments, each CLB output requires about 0.4 mW/MHz of its output frequency. Graphs of power versus operating frequency are shown in Table 1.

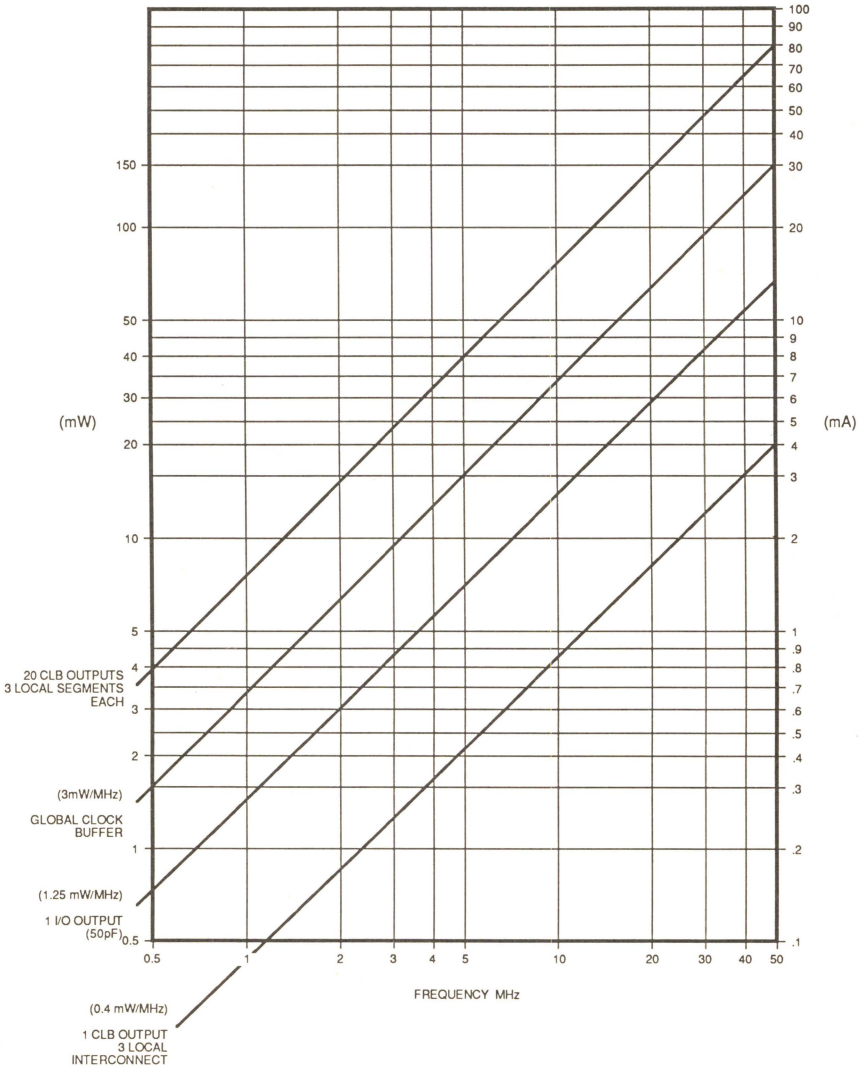



**Table 1. Typical LCA Power Consumption by Element**

## Programming

Configuration data to define the function and interconnection within a Logic Cell Array are loaded automatically at power-up or upon command. Several methods of automatically loading the required data are designed into the Logic Cell Array and are determined by logic levels applied to mode selection pins at configuration time. The form of the data may be either serial or parallel, depending on the configuration mode. The programming data are independent of the configuration mode selected. The state diagram of Figure 12 illustrates the configuration process.

Input thresholds for user I/O pins can be selected to be either TTL-compatible or CMOS-compatible and remain in that state until the LCA begins operation. If the user has selected CMOS compatibility, the input thresholds are changed to CMOS levels during configuration.

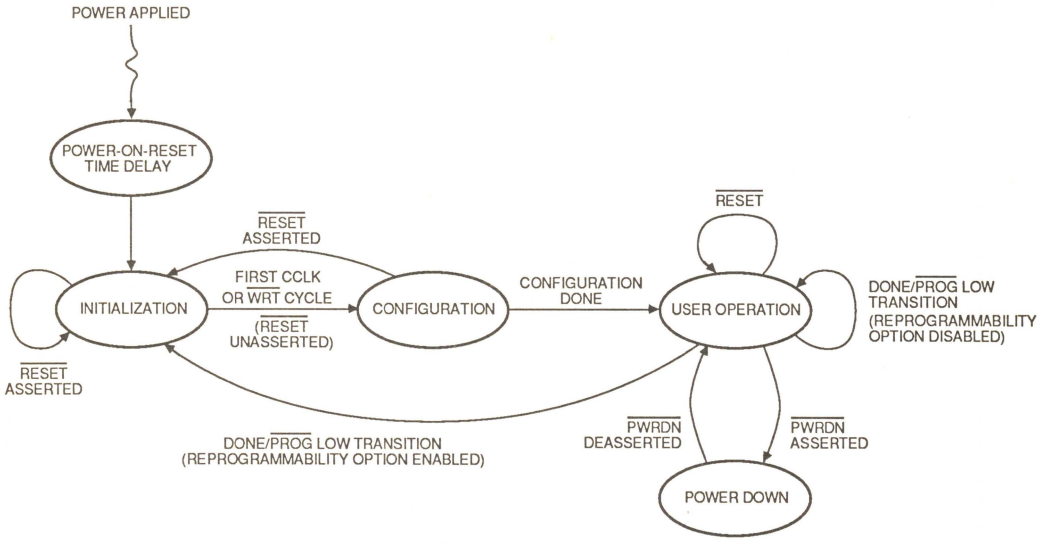



**Figure 12. Configuration State Diagram**

Figure 13 shows the specific data arrangement for the M2064 device. Future products will use the same data format to maintain compatibility between different devices of the Monolithic Memories' product line, but they will have different sizes and numbers of data frames. For the M2064 configuration requires 12,038 bits for each device. For the M2018, the configuration of each device requires 17,878 bits. The M2064 uses 160 configuration data frames and the M2018 uses 197.

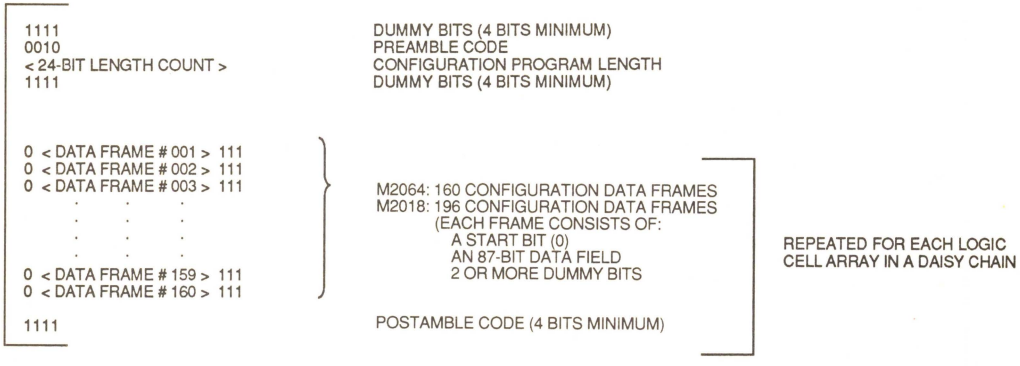



**Figure 13. M2064 Configuration Data Arrangement**

The configuration bit stream begins with preamble bits, a preamble code and a length count. The length count is loaded into the control logic of the Logic Cell Array and is used to determine the completion of the configuration process. When configuration is initiated, a 24-bit length counter is set to 0 and begins to count the total number of configuration clock cycles applied to the device. When the current length count equals the loaded length count, the configuration process is complete. Two clocks before completion, the internal logic becomes active and is reset. On the next clock, the inputs and outputs become active as configured and consideration should be given to avoid configuration signal contention. (Attention must be paid to avoid contention on pins which are used as inputs during configuration and become outputs in operation.) On the last configuration clock, the completion of configuration is signalled by the release of the DONE, $\overline{PROG}$ pin of the device as the device begins operation. This open-drain output can be AND-tied with multiple Logic Cell Arrays and used as an active-high READY or active-low, $\overline{RESET}$, to other portions of the system. High during configuration (HDC) and low during configuration ($\overline{LDC}$), are released one CCLK cycle before DONE is asserted. In master mode configurations, it is convenient to use $\overline{LDC}$ as an active-low EPROM chip enable.

As each data bit is supplied to the LCA, it is internally assembled into a data word. As each data word is completely assembled, it is loaded in parallel into one word of the internal configuration memory array. The last word must be loaded before the current length count compare is true. If the configuration data are in error, e.g., PROM address lines swapped, the LCA will not be ready at the length count and the counter will cycle through an additonal complete count prior to configuration being "done".

Figure 14 shows the selection of the configuration mode based on the state of the mode pins M0 and M1. These package pins are sampled prior to the start of the configuration process to determine the mode to be used. Once configuration is DONE and subsequent operation has begun, the mode pins may be used to perform data readback, as discussed later. An additional mode pin, M2, must be defined at the start of configuration. This package pin is a user-configurable I/O after configuration is complete.

| MODE PIN | | | MODE SELECTED |
|---|---|---|---|
| M0 | M1 | M2 | |
| 0 | 0 | 0 | Master serial |
| 0 | 0 | 1 | Master LOW mode |
| 0 | 1 | 1 | Master HIGH mode |
| 1 | 0 | 1 | Peripheral mode |
| 1 | 1 | 1 | Slave mode |

Master LOW addresses begin at 0000 and increment.
Master HIGH addresses begin at FFFF and decrement.

**Figure 14. Configuration Mode Selection**

**Initialization Phase**

When power is applied, an internal power-on-reset circuit is triggered. When VCC reaches the voltage at which the LCA begins to operate (2.5 to 3 Volts), the chip is initialized, outputs are made high-impedance and a time-out is initiated to allow time for power to stabilize. This time-out (15 to 35 ms) is determined by a counter driven by a self-generated, internal sampling clock that drives the configuration clock (CCLK) in master configuration mode. This internal sampling clock will vary with process, temperature and power supply over the range of 0.5 to 1.5 MHz. LCAs with mode lines set for master mode will time-out of their initialization using a longer counter (60 to 140 ms) to assure that all devices, which it may be driving in a daisy chain, will be ready. Configuration using peripheral or slave modes must be delayed long enough for this initialization to be completed.

The initialization phase may be extended by asserting the active-low external $\overline{RESET}$. If a configuration has begun, an assertion of $\overline{RESET}$ will initiate an abort, including an orderly clearing of partially loaded configuration memory bits. After about three clock cycles for synchronization, initialization will require about 160 additional cycles of the internal sampling clock (197 for the M2018) to clear the internal memory before another configuration may begin. The same is true of a configured part in which the reconfigurable control bit is set. When a HIGH-to-LOW transition on the DONE, $\overline{PROG}$ package pin is detected, thereby initiating a reprogram, the configuration memory is cleared. This insures an orderly configuration in which no internal signal conflicts are generated during the loading process.

**Master Mode**

In master mode, the Logic Cell Array automatically loads the configuration program from an external memory device. Figure 15a shows an example of the master mode connections required. The Logic Cell Array provides sixteen address outputs and the control signals $\overline{RCLK}$ (read clock), HDC (high during configuration) and $\overline{LDC}$ (low during configuration) to execute read cycles from the external memory. Parallel eight-bit data words are read and internally serialized. As each data word is read, the least significant bit of each byte, normally D0, is the next bit in the serial stream.

Addresses supplied by the Logic Cell Array can be selected by the mode lines to begin at address 0 and incremented to read the memory (master low mode), or they can begin at address FFFF Hex and be decremented (master high mode). This capability is provided to allow the Logic Cell Array to share external memory with another device, such as a microprocessor. For example, if the processor begins its execution from low memory, the Logic Cell Array can load itself from high memory and enable the processor to begin execution once configuration is completed. The DONE, $\overline{PROG}$ output pin can be used to hold the processor in a Reset state until the Logic Cell Array has completed the configuration process.

The master serial mode uses serial configuration data, synchronized by the rising edge of RCLK, as in Figure 15b.

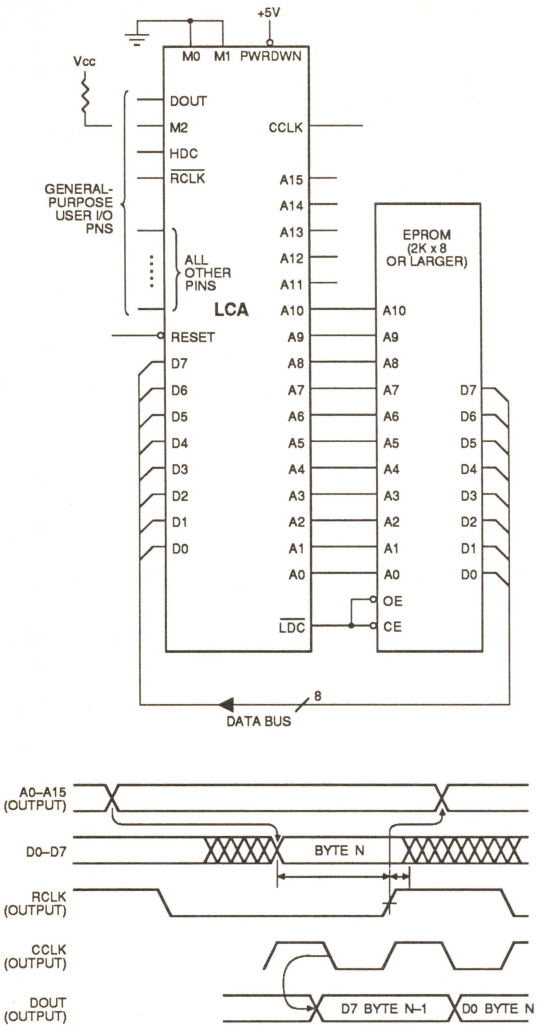



Figure 15a. Master Low Address Configuration

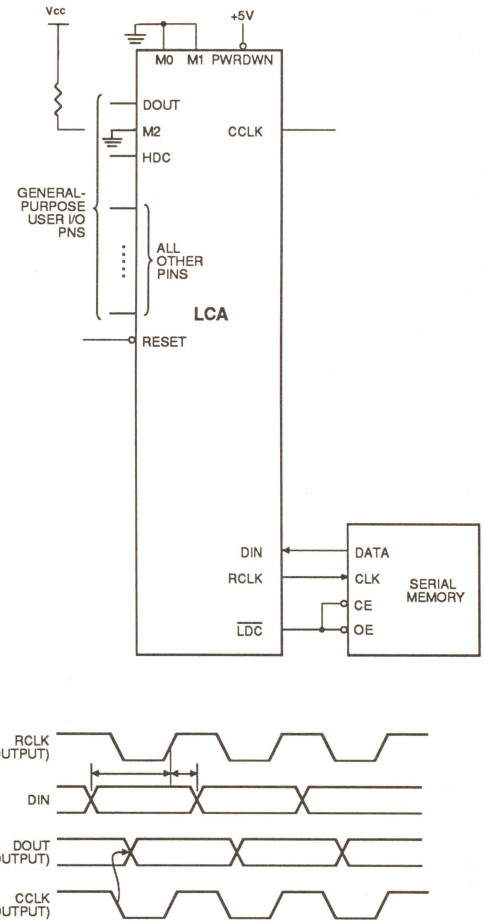



Figure 15b. Master Serial Mode Configuration

### Peripheral Mode

Peripheral mode provides a simplified interface through which the device may be loaded as a processor peripheral. Figure 16 shows the peripheral mode connections. Processor write cycles are decoded from the common assertion of the active-low write strobe ($\overline{WRT}$), and two active-low and one active-high chip selects ($\overline{CS0}$, $\overline{CS1}$, CS2). If all these signals are not available, the unused inputs should be driven to their respective active levels. The Logic Cell Array will accept one bit of the configuration program on the data input (DIN) pin for each processor write cycle. Data is supplied in the serial sequence described earlier.

Since only a single bit from the processor data bus is loaded per cycle, the loading process involves the processor reading a byte or word of data, writing a bit of the data to the Logic Cell Array, shifting the word and writing a bit until all bits of the word are written, then continuing in the same fashion with the next word, etc. After the configuration program has been loaded, an additional three clocks (a total of three more than the length count) must be supplied in order to complete the configuration process. When more than one device is being used in the system, each device can be assigned a different bit in the processor data bus, and multiple devices can be loaded on each processor write cycle. This "broadside" loading method provides a very easy and time-efficient method of loading several devices.

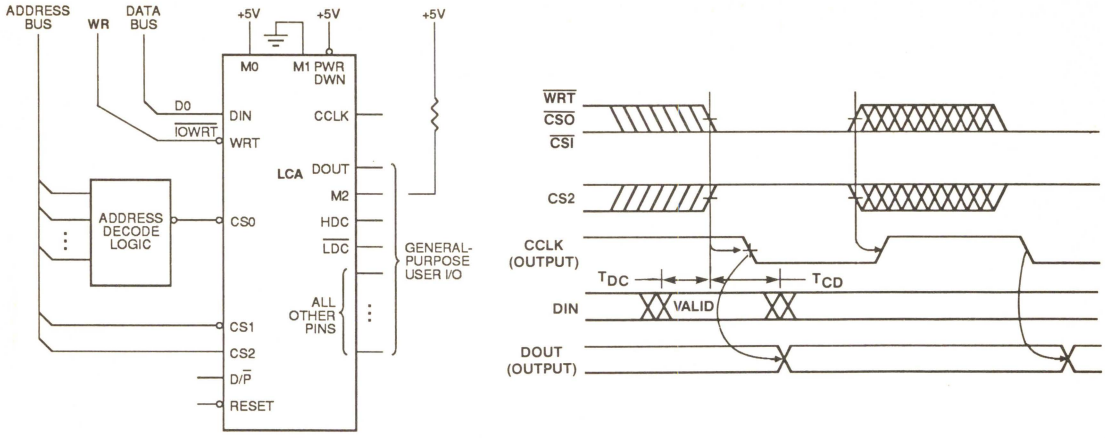



Figure 16. Peripheral Mode Configuration

### Slave Mode

Slave mode, Figure 17, provides the simplest interface for loading the Logic Cell Array configuration. Data is supplied in conjunction with a synchronizing clock. For each LOW-to-HIGH input transition of configuration clock (CCLK), the data present on the data input (DIN) pin is loaded into the internal shift register. Data may be supplied by a processor or by other special circuits. Slave mode is used for downstream devices in a daisy-chain configuration. The data for each slave LCA are supplied by the preceding LCA in the chain, and the clock is supplied by the lead device, which is configured in master or peripheral mode. After the configuration program has been loaded, an additional three clocks (a total of three more than the length count) must be supplied in order to complete the configuration process.

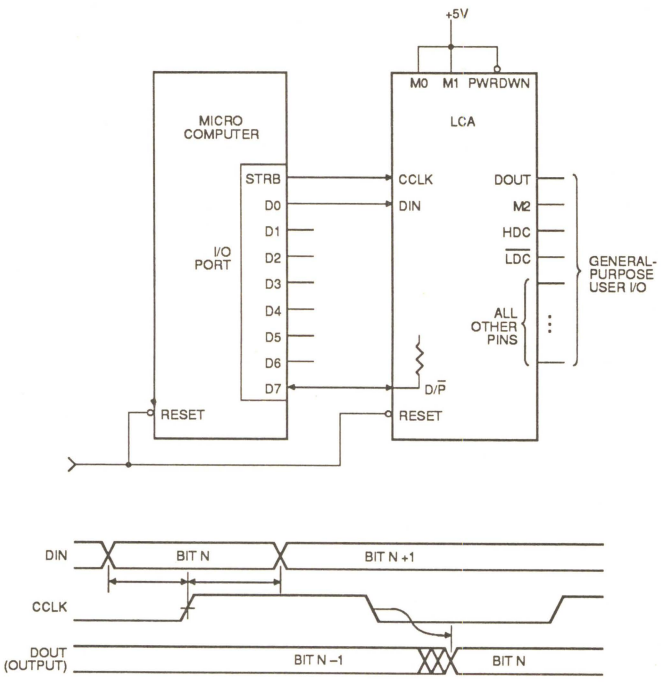



Figure 17. Slave Mode Configuration

### Daisy Chain

The daisy-chain programming mode is supported by Logic Cell Array in all programming modes. In master mode and peripheral mode, the LCA can act as a source of data and control for slave devices. For example, Figure 18 shows a single device in master mode, with two devices in slave mode. The master mode device reads the external memory and begins the configuration loading process for all of the devices.

The data begin with a preamble and a length count which is supplied to all devices at the beginning of the configuration. The length count represents the total number of cycles required to load all of the devices in the daisy chain. After loading the length count, the lead device will load its configuration data while providing a HIGH DOUT to downstream devices. When the lead device has been loaded and the current length count has not reached the full value, memory access continues. Data bytes are read and serialized by the lead device. The data are passed through the lead device and appear on the data out (DOUT) pin in serial form. The lead device also generates the configuration clock (CCLK) to synchronize the serial output data. A master mode device generates an internal CCLK of eight times the EPROM address rate, while a peripheral mode device produces CCLK from the chip select and write strobe timing.

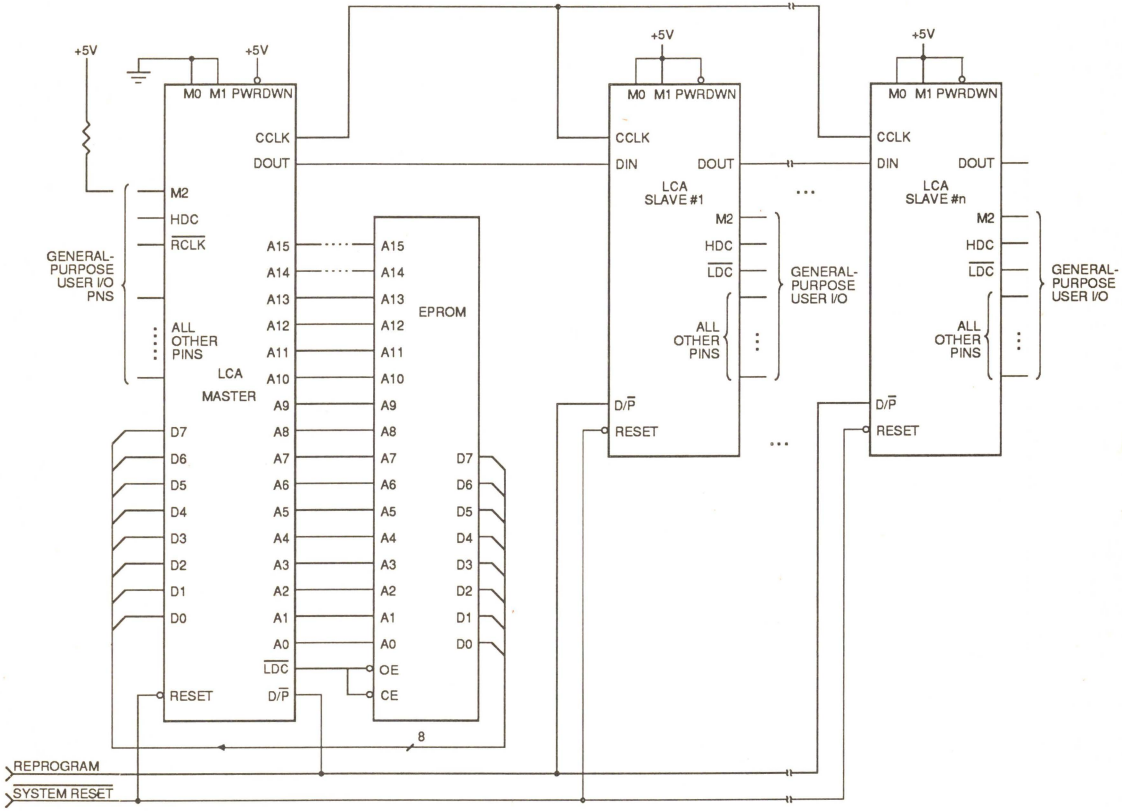



**Figure 18. Master Mode with Daisy Chain**

### Operation

When all of the devices have been loaded and the length count is complete, a synchronous start-up of operation is performed. On the clock cycle following the end of loading, the internal logic begins functioning in the reset state. On the next CCLK, the configured output buffers become active to allow signals to stabilize. The next CCLK cycle produces the DONE condition. The length count control of operation allows a system of multiple Logic Cell Arrays to begin operation in a synchronized fashion. If the crystal oscillator is used, it will begin operation before configuration is complete to allow time for stabilization before it is connected to the internal circuitry.

## Special Features

In addition to the normal user logic functions and interconnect, the configuration data include control for several special functions:

- Input thresholds
- Readback enable
- Reprogram enable
- DONE pull-up resistor

Each of these functions is controlled by a portion of the configuration program generated by the XACT Development System.

### Input Thresholds

During configuration, all input thresholds are TTL level. During configuration input thresholds are established as specified, either TTL or CMOS. The $\overline{\text{PWRDN}}$ input threshold is an exception; it is always a CMOS level input. The TTL threshold option requires additional power for threshold shifting.

### Readback

After a Logic Cell Array has been programmed, the configuration program may be read back from the device. Readback may be used for verification of configuration and as a method of determining the state of internal logic nodes during debugging. In applications in which the verification is not used, it may be desirable to limit access to the configuration data. Three readback options are provided: 'on command', 'only once', and 'never'. If 'on-command readback' is selected, the device will respond to all readback requests. If 'readback once' is selected, the device will respond only to the first readback request after programming is complete. Subsequent readback requests will be ignored. If 'readback never' is selected, the device will not respond to a readback command.

Readback is accomplished without the use of any of the user I/O pins; only M0, M1, and CCLK pins are used. An initiation of readback is produced by a LOW-to-HIGH transition of the M0, RTRIG (read trigger) pin. Once the readback command has been given, CCLK is cycled to read back each data bit in a format similar to loading. After two dummy bits, the first data frame is shifted out, in inverted sense, on the M1, $\overline{\text{RDATA}}$ (read data) pin. All data frames must be read back to complete the process and return the mode select and CCLK pins to their normal functions.

In addition to the configuration program, the readback includes the current state of each of the internal logic block storage elements, and the state of the input (I) connection pin on each I/O block. This state information is used by the Logic Cell Array development system In-Circuit Emulator to provide visibility into the internal operation of the logic while the system is operating. To readback a uniform time sample of all storage elements it may be necessary to inhibit the system clock.

### Reprogram

The configuration memory of the Logic Cell Array may be rewritten while the device is in the user's system, if that option is selected when the LCA is configured. If another programming cycle is to be initiated, the dual function package pin DONE, $\overline{\text{PROG}}$ must be given a HIGH-to-LOW transition. Sensitivity to noise is reduced, by confirming the HIGH-to-LOW transition over two to three cycles using the LCA's internal sampling oscillator. When a reprogram command is recognized, all internal logic and connectivity definitions are erased and the I/O package pins are forced to a high impedance condition. The device returns to the initialization state. Reprogram control is often implemented with an external open collector driver which pulls DONE, $\overline{\text{PROG}}$ LOW. Once it recognizes a stable request, the Logic Cell Array will hold a LOW until the new configuration has been completed. Whether or not the reprogram request is maintained, the Logic Cell Array will begin operation upon completion of configuration.

### DONE Pull-up

The DONE, $\overline{\text{PROG}}$ pin is an open drain I/O that indicates programming status. As an input, it initiates a reprogram operation. An optional internal pull-up resistor may be enabled.

### Battery Backup

Because the control store of the Logic Cell Array is a CMOS static memory, its cells require only a very low standby current for data retention. In some systems, this low data retention current characteristic facilitates preserving configurations in the event of a primary power loss. The Logic Cell Array has built in power-down logic which, when activated, will disable normal operation of the device and retain only the configuration data. All internal operation is suspended and output buffers are placed in their high-impedance state.

Power-down data retention is possible with a simple battery-backup circuit because the power requirement is extremely low. For retention at 2.0 V, the required current is typically on the order of 0.5 mA. Screening of this parameter is available. To force the Logic Cell Array into the power-down state, the user must pull the $\overline{\text{PWRDWN}}$ pin low and continue to supply a retention voltage to the VCC pins of the package. When normal power is restored, VCC is elevated to its normal operating voltage and $\overline{\text{PWRDWN}}$ is returned to a HIGH. The Logic Cell Array resumes operation with the same internal sequence that occurs at the conclusion of configuration. Internal I/O and logic block storage elements will be reset, the outputs will become enabled and then the DONE, $\overline{\text{PROG}}$ pin will be released. No configuration programming is involved.

## Performance

The high performance of the Logic Cell Array results from its patented architectural features and from the use of an advanced high-speed CMOS manufacturing process. Performance may be measured in terms of minimum propagation times for logic elements.

Flip-flop loop delays for the I/O block and logic block flip-flops are about 3 ns. This short delay provides very good performance under asynchronous clock and data conditions. Short loop delays minimize the probability of a metastable condition which can result from assertion of the clock during data transitions. Because of the short loop delay characteristic in the Logic Cell Array, the I/O block flip-flops can be used very effectively to synchronize external signals applied to the device. Once synchronized in the I/O block, the signals can be used internally without further consideration of their clock relative timing, except as it applies to the internal logic and routing path delays.

### Device Performance

The single parameter which most accurately describes the overall performance of the Logic Cell Array is the maximum toggle rate for a logic block storage element configured as a toggle flip-flop. The configuration for determining the toggle performance of the Logic Cell Array is shown in Figure 19. The clock for the storage element is provided by the global clock buffer and the flip-flop output Q is fed back through the combinatorial logic to form the data input for the next clock edge. Using this arrangement, flip-flops in the Logic Cell Array can be toggled at clock rates from 33-70 MHz, depending on the speed grade used.

Actual Logic Cell Array performance is determined by the critical path speed, including both the speed of the logic and storage elements in that path, and the speed of the particular network routing. Figure 20 shows a typical system logic configuration of two flip-flops with an extra combinatorial level between them. Depending on speed grade, system clock rates to 35 MHz are practical for this logic. To allow the user to make the best use of the capabilities of the device, the delay calculator in the XACT Development System determines worst-case path delays using actual impedance and loading information.

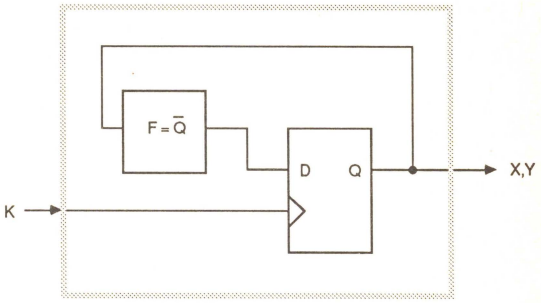



**Figure 19. Logic Block Configuration for Toggle Rate Measurement**

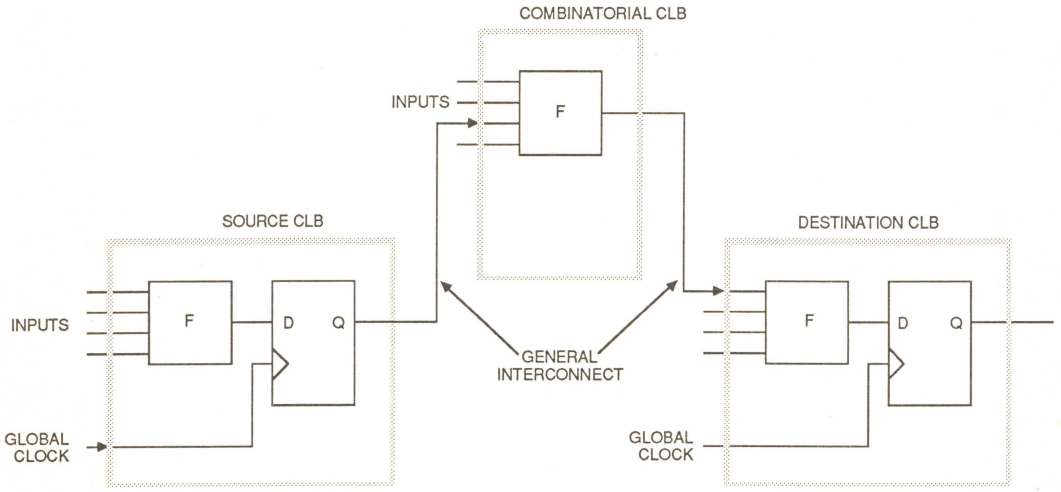



**Figure 20. Typical Logic Path**

### Logic Block Performance

Logic Block propagation times are measured from the interconnect point at the input of the combinatorial logic to the output of the block in the interconnect area. Combinatorial performance is independent of logic function because of the table look-up based implementation. Timing is different when the combinatorial logic is used in conjunction with the storage element. For the combinatorial logic function driving the data input of the storage element, the critical timing is data set-up relative to the clock edge provided to the storage element. The delay from the clock source to the output of the logic block is critical in the timing of signals produced by storage elements. The loading on a logic block output is limited only by the additional propagation delay of the interconnect network. Performance of the logic block is a function of supply voltage and temperature, as shown in Figures 21 and 22.

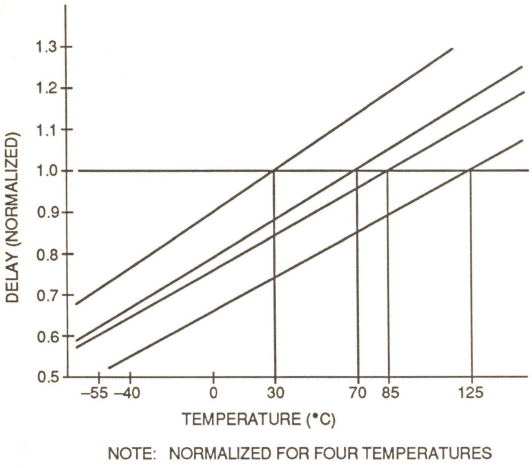



**Figure 21. Delay vs. Temperature**

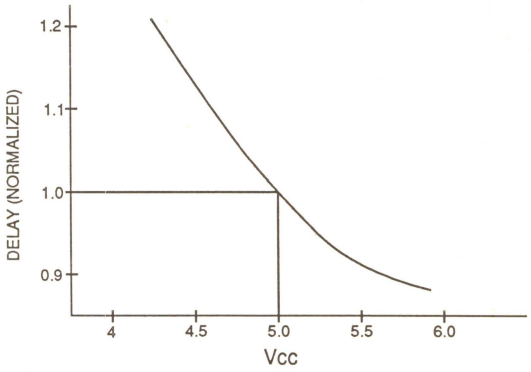



**Figure 22. Delay vs. Power Supply**

### Interconnect Performance

Interconnect performance depends on the routing resource used to implement the signal path. As discussed earlier, direct interconnect from block to block provides a minimum delay path for a signal.

The single metal segment used for long lines exhibits low resistance from end to end, but relatively high capacitance. Signals driven through a programmable switch will have the additional impedance of the switch added to their normal drive impedance.

General-purpose interconnect performance depends on the number of switches and segments used, the presence of the bidirectional repowering buffers and the overall loading on the signal path at all points along the path. In calculating the worst-case delay for a general interconnect path, the delay calculator portion of the XACT development system accounts for all of these elements. As an approximation, interconnect delay is proportional to the summation of totals of local metal segments beyond each programmable switch. In effect, the delay is a sum of R-C delays each approximated by an R times the total C it drives. The R of the switch and the C of the interconnect are functions of the particular device performance grade. For a string of three local interconnects, the approximate delay at the first segment, after the first switch resistance, would be three units; an additonal two delay units after the next switch plus an additional delay after the last switch in the chain. The interconnect R-C chain terminates at each repowering buffer. Nearly all of the capacitance is in the interconnect metal and switches; the capacitance of the block inputs is not significant. Figure 23 shows an estimation of this delay.

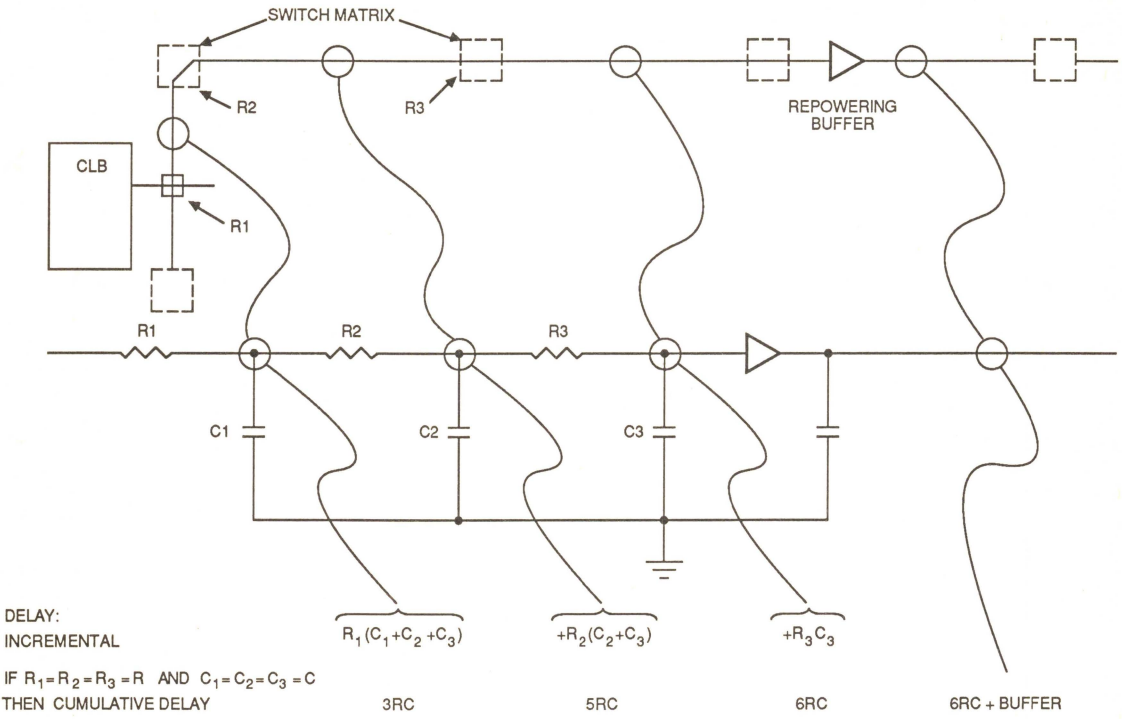



**Figure 23. Interconnection Delay Example**

5

## Development System

To support designers using the Logic Cell Array , Monolithic Memories provides a basic development system with several options for additional productivity. The XACT system provides the following:

- Graphic-driven design entry
- Schematic entry
- Interactive timing delay calculations
- Macrocell library support, both for standard Monolithic Memories supplied functions and user-defined functions
- Design entry checking for consistency and completeness
- Automatic design documentation generation
- Automatic placement and routing
- Simulation interface support, including automatic netlist (circuit description) and timing extraction
- In-circuit emulation for multiple devices

The host system on which the XACT system operates is an IBM™ PC-XT™ or PC-AT™ or compatible system with MS-DOS™ 2.1 or higher. Color graphics is required as well as 640 K bytes of internal RAM (an Expanded Memory Specification (EMS) card with 256 K bytes of memory is required for the M2018). A complete system requires one parallel I/O port and two serial ports and a mouse.

For more detailed information of the XACT Development System, please refer to Logic Cell Array Development System Datasheet.

| 48-PIN DIP | 68-PIN PLCC | 68-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ | | | | USER OPERATION |
|---|---|---|---|---|---|---|---|
| | | | SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | |
| | 1 | B6 | GND | | | | |
| | 2 | A6 | ≪HIGH≫ | | A13 (O) | | I/O |
| 1 | 3 | B5 | | | A6 (O) | | |
| | 4 | A5 | | | A12 (O) | | |
| 2 | 5 | B4 | | | A7 (O) | | |
| 3 | 6 | A4 | | | A11 (O) | | |
| 4 | 7 | B3 | | | A8 (O) | | |
| 5 | 8 | A3 | | | A10 (O) | | |
| 6 | 9 | A2 | | | A9 (O) | | |
| 7 | 10 | B2 | $\overline{\text{PWRDWN}}$ (I) | | | | |
| 8 | 11 | B1 | ≪HIGH≫ | | | | I/O |
| | 12 | C2 | | | | | |
| 9 | 13 | C1 | | | | | |
| | 14 | D2 | | | | | |
| 10 | 15 | D1 | | | | | |
| | 16 | E2 | | | | | |
| 11 | 17 | E1 | | | | | |
| 12 | 18 | F2 | VCC | | | | |
| 13 | 19 | F1 | ≪HIGH≫ | | | | I/O |
| | 20 | G2 | | | | | |
| 14 | 21 | G1 | | | | | |
| | 22 | H2 | | | | | |
| 15 | 23 | H1 | | | | | |
| 16 | 24 | J2 | | | | | |
| 17 | 25 | J1 | M1 (HIGH) | M1 (LOW) | M1 (HIGH) | M1 (LOW) | $\overline{\text{RDATA}}$ (O) |
| 18 | 26 | K1 | M0 (HIGH) | M0 (HIGH) | M0 (LOW) | M0 (LOW) | RTRIG (I) |
| 19 | 27 | K2 | M2 (HIGH) | | | | I/O |
| 20 | 28 | L2 | HDC (HIGH) | | | | |
| | 29 | K3 | ≪HIGH≫ | | | | |
| 21 | 30 | L3 | LDC (LOW) | | | | |
| | 31 | K4 | ≪HIGH≫ | | | | |
| 22 | 32 | L4 | | | | | |
| | 33 | K5 | | | | | |
| 23 | 34 | L5 | | | | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 2a. M2064 Pin Assignments**

**(continued on next page)**

| 48-PIN DIP | 68-PIN PLCC | 68-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ | | | | USER OPERATION |
|---|---|---|---|---|---|---|---|
| | | | SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | |
| 24 | 35 | K6 | GND | | | | |
| | 36 | L6 | ≪HIGH≫ | | | | I/O |
| 25 | 37 | K7 | | | | | |
| | 38 | L7 | | | | | |
| 26 | 39 | K8 | | | | | |
| 27 | 40 | L8 | | | | | |
| 28 | 41 | K9 | | | D7 (I) | | |
| 29 | 42 | L9 | | | D6 (I) | | |
| 30 | 43 | L10 | | | | | XTL2 or I/O |
| 31 | 44 | K10 | $\overline{\text{RESET}}$ (I) | | | | |
| 32 | 45 | K11 | DONE (O) | | | | $\overline{\text{PROG}}$ (I) |
| 33 | 46 | J10 | ≪HIGH≫ | | | | XTL1 or I/O |
| | 47 | J11 | | | | | I/O |
| 34 | 48 | H10 | | | D5 (I) | | |
| | 49 | H11 | | | | | |
| 35 | 50 | G10 | | $\overline{\text{CS0}}$ (I) | D4 (I) | | |
| 36 | 51 | G11 | | $\overline{\text{CS1}}$ (I) | D3 (I) | | |
| | 52 | F10 | VCC | | | | |
| | 53 | F11 | ≪HIGH≫ | | | | I/O |
| 37 | 54 | E10 | | CS2 (I) | D2 (I) | | |
| | 55 | E11 | | | | | |
| 38 | 56 | D10 | | $\overline{\text{WRT}}$ (I) | D1 (I) | | |
| 39 | 57 | D11 | | | $\overline{\text{RCLK}}$ | | |
| 40 | 58 | C10 | DIN (I) | | D0 (I) | | |
| 41 | 59 | C11 | DOUT (O) | | | | |
| 42 | 60 | B11 | CCLK (I) | CCLK (O) | | | CCLK (I) |
| 43 | 61 | B10 | ≪HIGH≫ | | A0 (O) | | I/O |
| 44 | 62 | A10 | | | A1 (O) | | |
| 45 | 63 | B9 | | | A2 (O) | | |
| 46 | 64 | A9 | | | A3 (O) | | |
| | 65 | B8 | | | A15 (O) | | |
| 47 | 66 | A8 | | | A4 (O) | | |
| | 67 | B7 | | | A14 (O) | | |
| 48 | 68 | A7 | | | A5 (O) | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 2a. M2064 Pin Assignments (continued)**

| 68-PIN PLCC | 68-PIN PGA | 84-PIN PLCC | 84-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ | | | | USER OPERATION |
|---|---|---|---|---|---|---|---|---|
| | | | | SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | |
| 1 | B6 | 1 | C6 | GND | | | | |
| 2 | A6 | 2 | A6 | ≪HIGH≫ | | A13 (O) | | I/O |
| | | 3 | A5 | | | | | |
| | | 4 | B5 | | | | | |
| 3 | B5 | 5 | C5 | | | A6 (O) | | |
| 4 | A5 | 6 | A4 | | | A12 (O) | | |
| 5 | B4 | 7 | B4 | | | A7 (O) | | |
| 6 | A4 | 8 | A3 | | | A11 (O) | | |
| 7 | B3 | 9 | A2 | | | A8 (O) | | |
| 8 | A3 | 10 | B3 | | | A10 (O) | | |
| 9 | A2 | 11 | A1 | | | A9 (O) | | |
| 10 | B2 | 12 | B2 | $\overline{\text{PWRDWN}}$ (I) | | | | |
| 11 | B1 | 13 | C2 | ≪HIGH≫ | | | | I/O |
| 12 | C2 | 14 | B1 | | | | | |
| 13 | C1 | 15 | C1 | | | | | |
| 14 | D2 | 16 | D2 | | | | | |
| 15 | D1 | 17 | D1 | | | | | |
| | | 18 | E3 | | | | | |
| 16 | E2 | 19 | E2 | | | | | |
| | | 20 | E1 | | | | | |
| 17 | E1 | 21 | F2 | | | | | |
| 18 | F2 | 22 | F3 | VCC | | | | |
| 19 | F1 | 23 | G3 | ≪HIGH≫ | | | | I/O |
| | | 24 | G1 | | | | | |
| 20 | G2 | 25 | G2 | | | | | |
| | | 26 | F1 | | | | | |
| 21 | G1 | 27 | H1 | | | | | |
| 22 | H2 | 28 | H2 | | | | | |
| 23 | H1 | 29 | J1 | | | | | |
| 24 | J2 | 30 | K1 | | | | | |
| 25 | J1 | 31 | J2 | M1 (HIGH) | M1 (LOW) | M1 (HIGH) | M1 (LOW) | $\overline{\text{RDATA}}$ (O) |
| 26 | K1 | 32 | L1 | M0 (HIGH) | M0 (HIGH) | M0 (LOW) | M0 (LOW) | RTRIG (I) |
| 27 | K2 | 33 | K2 | M2 (HIGH) | | | | I/O |
| 28 | L2 | 34 | K3 | HDC (HIGH) | | | | |
| 29 | K3 | 35 | L2 | ≪HIGH≫ | | | | |
| 30 | L3 | 36 | L3 | LDC (LOW) | | | | |
| 31 | K4 | 37 | K4 | ≪HIGH≫ | | | | |
| 32 | L4 | 38 | L4 | | | | | |
| | | 39 | J5 | | | | | |
| 33 | K5 | 40 | K5 | | | | | |
| 34 | L5 | 41 | L5 | | | | | |
| | | 42 | K6 | | | | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 2b. M2018 Pin Assignments (continued on next page)**

| 68-PIN PLCC | 68-PIN PGA | 84-PIN PLCC | 84-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | USER OPERATION |
|---|---|---|---|---|---|---|---|---|
| 35 | K6 | 43 | J6 | GND | | | | |
| | | 44 | J7 | ≪HIGH≫ | | | | I/O |
| 36 | L6 | 45 | L7 | | | | | |
| 37 | K7 | 46 | K7 | | | | | |
| 38 | L7 | 47 | L6 | | | | | |
| | | 48 | L8 | | | | | |
| 39 | K8 | 49 | K8 | | | | | |
| 40 | L8 | 50 | L9 | | | | | |
| 41 | K9 | 51 | L10 | | | D7 (I) | | |
| 42 | L9 | 52 | K9 | | | D6 (I) | | |
| 43 | L10 | 53 | L11 | | | | | XT2 or I/O |
| 44 | K10 | 54 | K10 | $\overline{\text{RESET}}$ (I) | | | | |
| 45 | K11 | 55 | J10 | DONE (0) | | | | $\overline{\text{PROG}}$ (1) |
| 46 | J10 | 56 | K11 | ≪HIGH≫ | | | | XTL1 or I/O |
| 47 | J11 | 57 | J11 | | | | | I/O |
| 48 | H10 | 58 | H10 | | | D5 (I) | | |
| | | 59 | H11 | | | | | |
| 49 | H11 | 60 | F10 | | | | | |
| | | 61 | G10 | | | | | |
| 50 | G10 | 62 | G11 | | $\overline{\text{CS0}}$ (I) | D4 (I) | | |
| 51 | G11 | 63 | G9 | | $\overline{\text{CS1}}$ (I) | D3 (I) | | |
| 52 | F10 | 64 | F9 | VCC | | | | |
| 53 | F11 | 65 | F11 | ≪HIGH≫ | | | | I/O |
| 54 | E10 | 66 | E11 | | CS2 (I) | D2 (I) | | |
| | | 67 | E10 | | | | | |
| 55 | E11 | 68 | E9 | | | | | |
| | | 69 | D11 | | | | | |
| 56 | D10 | 70 | D10 | | $\overline{\text{WRT}}$ (I) | D1 (I) | | |
| 57 | D11 | 71 | C11 | | | $\overline{\text{RCLK}}$ | | |
| 58 | C10 | 72 | B11 | DIN (I) | | D0 (I) | | |
| 59 | C11 | 73 | C10 | DOUT (O) | | | | |
| 60 | B11 | 74 | A11 | CCLK (I) | CCLK (O) | | | CCLK (I) |
| 61 | B10 | 75 | B10 | ≪HIGH≫ | | A0 (O) | | I/O |
| 62 | A10 | 76 | B9 | | | A1 (O) | | |
| 63 | B9 | 77 | A10 | | | A2 (O) | | |
| 64 | A9 | 78 | A9 | | | A3 (O) | | |
| 65 | B8 | 79 | B8 | | | A15 (O) | | |
| 66 | A8 | 80 | A8 | | | A4 (O) | | |
| 67 | B7 | 81 | B6 | | | A14 (O) | | |
| | | 82 | B7 | | | | | |
| | | 83 | A7 | | | | | |
| 68 | A7 | 84 | C7 | | | A5 (O) | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 2b. M2018 Pin Assignments (continued)**

## Absolute Maximum Ratings*

Supply voltage $V_{CC}$ ..... -0.5 V to 7 V
Power down $V_{CC}$ ..... 2 V to 7 V
Input voltage ..... -0.5 V to $V_{CC}$ 0.5 V
Voltage applied to three-state output ..... -0.5 V to $V_{CC}$ 0.5 V
Storage temperature range ..... -65°C to +150°C
Lead temperature (soldering, 10 seconds) ..... 260°C

* Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those listed under "Recommended Operating Conditions" is not implied. Exposure to "Absolute Maximum Ratings" conditions for extended periods of time may affect device reliability.

## Operating Conditions

| SYMBOL | PARAMETER | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage relative to GND | 4.75 | | 5.25 | V |
| $V_{IHT}$ | High level input voltage—TTL configuration | 2.0 | | $V_{CC}$ | V |
| $V_{IHC}$ | High level input voltage—CMOS configuration | 0.7 $V_{CC}$ | | $V_{CC}$ | V |
| $V_{ILT}$ | Low level input voltage—TTL configuration | 0 | | 0.8 | V |
| $V_{ILC}$ | Low level input voltage—CMOS configuration | 0 | | 0.2 $V_{CC}$ | V |
| $I_{IT}$ | Input leakage current—TTL configuration | ±10 | | | μA |
| $I_{IC}$ | Input leakage current—CMOS configuration | ±10 | | | μA |
| $I_{OZ}$ | Three-state output off current ($V_{CC}$ = 5.5 V) | ±10 | | | μA |
| $t_{OP}$ | Operating free-air temperature | 0 | | 70 | °C |

## Electrical Characteristics Over Operating Conditions

| SYMBOL | PARAMETER | | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | High level output voltage | | $V_{CC}$ = 4.75 V | $I_{OH}$ = -4.0 mA | 3.86 | | | V |
| $V_{OL}$ | Low level output voltage | | $V_{CC}$ = 4.75 V | $I_{OL}$ = 4.0 mA | | | 0.32 | V |
| $I_{CCO}$ | Quiescent operating power supply current | CMOS inputs | $V_{CC}$ = 5.0 V | | | | 5 | mA |
| | | TTL inputs | $V_{CC}$ = 5.0 V | | | | 10 | mA |
| $I_{CCPD}$ | Power down supply current | | $V_{CC}$ = 5.0 V | | | 0.5 | | mA |

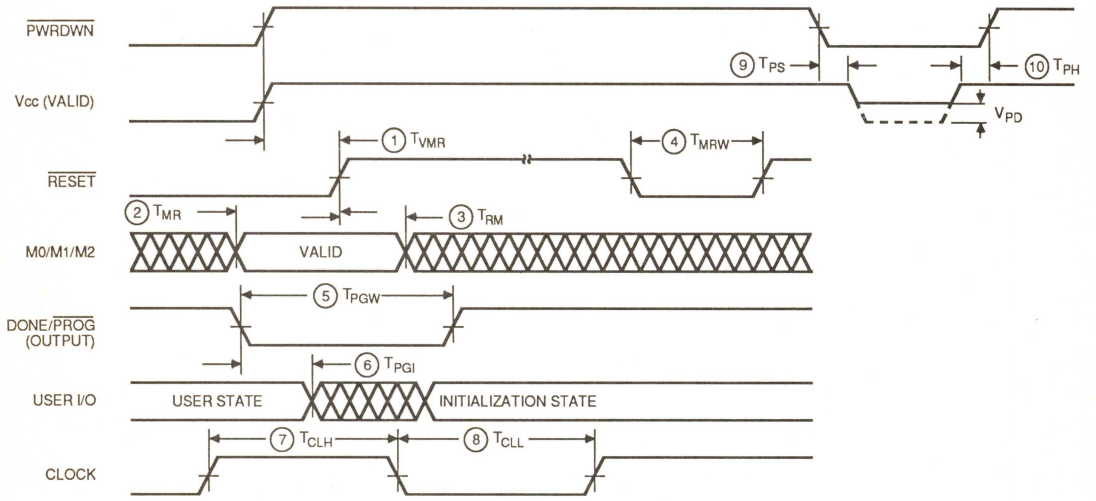
## Power On Timing

The LCAs contain on-chip reset timing logic for power-up operation. To insure proper master mode system operation, VCC must rise from 2.0 V to minimum specification level in 10 ms or less. For other modes, initiation of configuration must be delayed for 60 ms after VCC reaches the minimum specified level.

## Switching Characteristics — General

| SYMBOL | DESCRIPTION | | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{VMR}$ ① | $\overline{RESET}$ (2) | $V_{CC}$ setup (2.0 V) | 150 | | 150 | | 150 | | ns |
| $t_{MR}$ ② | | M2, M1, M0 setup | 60 | | 60 | | 60 | | ns |
| $t_{RM}$ ③ | | M2, M1, M0 hold | 60 | | 60 | | 60 | | ns |
| $t_{MRW}$ ④ | | Width (LOW) | 150 | | 150 | | 150 | | ns |
| $t_{PGW}$ ⑤ | DONE/$\overline{PROG}$ | Program width (LOW) | 6 | | 6 | | 6 | | μs |
| $t_{PGI}$ ⑥ | | Initialization | | 7 | | 7 | | 7 | μs |
| $t_{CLH}$ ⑦ | CLOCK | Clock (HIGH) | 12 | | 8 | | 7 | | ns |
| $t_{CLL}$ ⑧ | | Clock (LOW) | 12 | | 8 | | 7 | | ns |
| $t_{PS}$ ⑨ | $\overline{PWR}$ $\overline{DWN}$ | Setup to $V_{CC}$ | 0 | | 0 | | 0 | | ns |
| $t_{PH}$ ⑩ | | Hold from $V_{CC}$ | 0 | | 0 | | 0 | | ns |
| $V_{PD}$ | | Power Down | 2.0 | | 2.0 | | 2.0 | | V |

Notes: 1. $V_{CC}$ must rise from 2.0 Volts to $V_{CC}$ minimum in lest than 10 ms for master mode.
2. $\overline{RESET}$ timing relative to power-on and valid mode lines (M0, M1, M2) is relevant only when $\overline{RESET}$ is used to delay configuration.
3. Minimum CLOCK widths for the auxilliary buffer are 1.25 times the $t_{CLH}$, $t_{CLL}$.




5

## Switching Characteristics — CLB

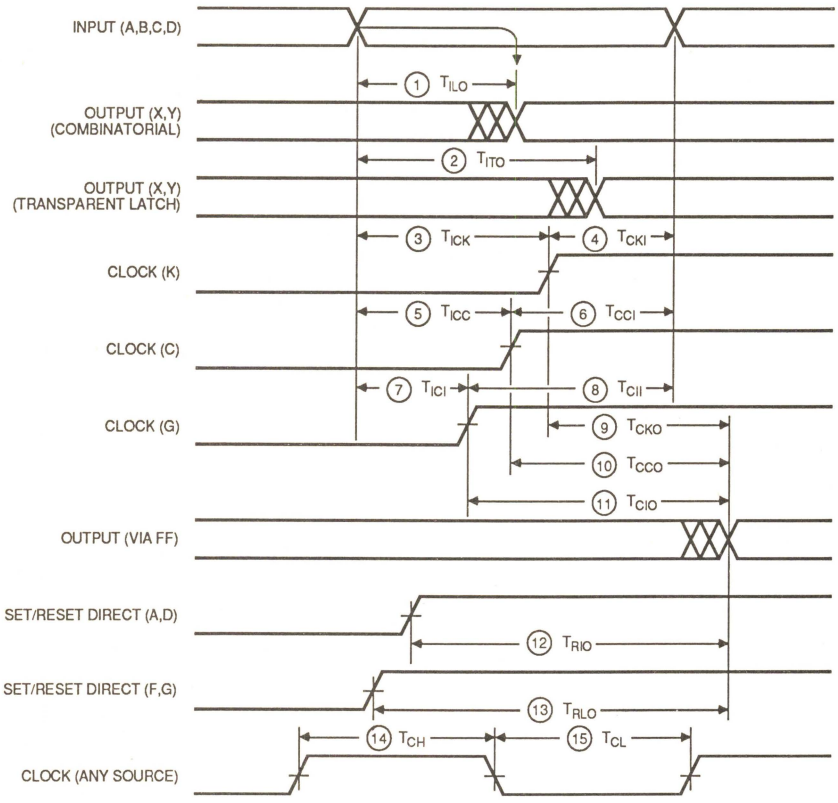
| SYMBOL | DESCRIPTION | | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{ILO}$ ① | Logic input to output | Combinatorial | | 20 | | 15 | | 10 | ns |
| $t_{ITO}$ ② | | Transparent latch | | 25 | | 20 | | 14 | ns |
| $t_{QLO}$ | | Additional for Q through F or G to out | | 13 | | 8 | | 6 | ns |
| $t_{CKO}$ ⑨ | K Clock | To output | | 20 | | 15 | | 10 | ns |
| $t_{ICK}$ ③ | | Logic-input setup | 12 | | 8 | | 7 | | ns |
| $t_{CKI}$ ④ | | Logic-input hold | 0 | | 0 | | 0 | | ns |
| $t_{CCO}$ ⑩ | C Clock | To output | | 25 | | 19 | | 13 | ns |
| $t_{ICC}$ ⑤ | | Logic-input setup | 12 | | 9 | | 6 | | ns |
| $t_{CCI}$ ⑥ | | Logic-input hold | 6 | | 0 | | 0 | | ns |
| $t_{CIO}$ ⑪ | Logic input to G Clock | To output | | 37 | | 27 | | 20 | ns |
| $t_{ICI}$ ⑦ | | Logic-input setup | 6 | | 4 | | 3 | | ns |
| $t_{CII}$ ⑧ | | Logic-input hold | 9 | | 5 | | 4 | | ns |
| $t_{RIO}$ ⑫ | Set/reset direct | Input A or D to out | | 25 | | 22 | | 16 | ns |
| $t_{RLO}$ ⑬ | | Through F or G to out | | 37 | | 28 | | 21 | ns |
| $t_{MRQ}$ | | Master Reset pin to out | | 35 | | 25 | | 20 | ns |
| $t_{RS}$ | | Separation of set/reset | 17 | | 9 | | 7 | | ns |
| $t_{RPW}$ | | Set/reset pulse-width | 12 | | 9 | | 7 | | ns |
| $F_{CLK}$ | Flip-flop toggle rate | Q through F to flip-flop | 33 | | 50 | | 70 | | MHz |
| $t_{CH}$ ⑭ | Clock | Clock HIGH | 12 | | 8 | | 7 | | ns |
| $t_{CL}$ ⑮ | | Clock LOW | 12 | | 8 | | 7 | | ns |

Note: All switching characteristics apply to all valid combinations of process, temperature and supply.

## Cross Reference Guide

| XILINX | MMI | $V_{CC}$ | | $F_{MAX}$ |
|---|---|---|---|---|
| | | MIN | MAX | MIN |
| XC2064-1 | | 4.5 V | 5.5 V | 20 MHz |
| | M2064-20 | 4.75 V | 5.25 V | 20 MHz |
| XC2064-2 | | 4.5 V | 5.5 V | 33 MHz |
| XC2064-33 | M2064-33 | 4.75 V | 5.25 V | 33 MHz |
| XC2064-50 | M2064-50 | 4.75 V | 5.25 V | 50 MHz |
| XC2064-70 | M2064-70 | 4.75 V | 5.25 V | 70 MHz |
| XC2018-33 | M2018-33 | 4.75 V | 5.25 V | 33 MHz |
| XC2018-50 | M2018-50 | 4.75 V | 5.25 V | 50 MHz |
| XC2018-70 | M2018-70 | 4.75 V | 5.25 V | 70 MHz |

## Switching Characteristics CLB

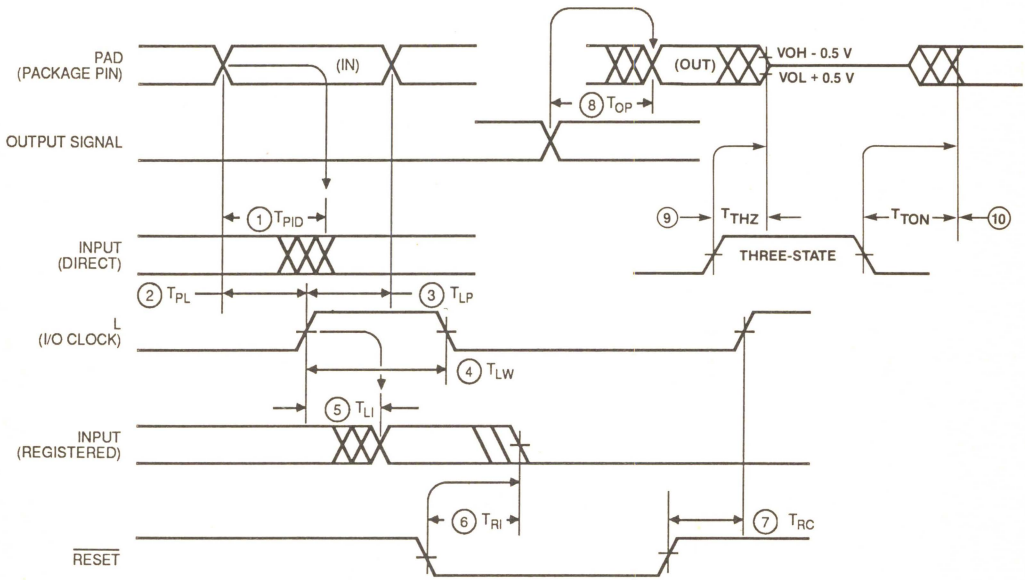



5

## Switching Characteristics — IOB

| SYMBOL | DESCRIPTION | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{PID}$ ① | Pad (package pin) to input (direct) | | 12 | | 8 | | 6 | ns |
| $t_{LI}$ ⑤ | I/O Clock to input (storage) | | 20 | | 15 | | 11 | ns |
| $t_{PL}$ ② | I/O Clock to pad-input setup | 12 | | 8 | | 6 | | ns |
| $t_{LP}$ ③ | I/O Clock to pad-input hold | 0 | | 0 | | 0 | | ns |
| $t_{LW}$ ④ | I/O Clock pulse width | 12 | | 9 | | 7 | | ns |
| | I/O Clock frequency | 33 | | 50 | | 70 | | MHz |
| $t_{OP}$ ⑧ | Output to pad (output enabled) | | 15 | | 12 | | 9 | ns |
| $t_{THZ}$ ⑨ | Three-state to pad begin hi-Z | | 25 | | 20 | | 15 | ns |
| $t_{TON}$ ⑩ | Three-state to pad end hi-Z | | 25 | | 20 | | 15 | ns |
| $t_{RI}$ ⑥ | $\overline{\text{RESET}}$ to input (storage) | | 40 | | 30 | | 25 | ns |
| $t_{RC}$ ⑦ | $\overline{\text{RESET}}$ to input clock | | 35 | | 25 | | 20 | ns |

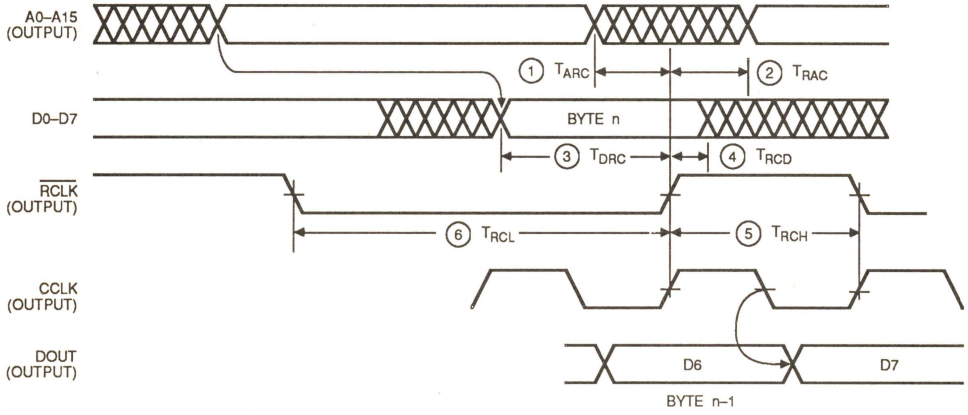
Note: Timing is measured at 0.5 $V_{CC}$ levels with 50 pF output load.



## Switching Characteristics — Programming — Master Mode

| SYMBOL | DESCRIPTION | | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{ARC}$ ① | RCLK | From address invalid | | 0 | | 0 | | 0 | ns |
| $t_{RAC}$ ② | | To address valid | | 200 | | 200 | | 200 | ns |
| $t_{DRC}$ ③ | | To data setup | 60 | | 60 | | 60 | | ns |
| $t_{RCD}$ ④ | | To data hold | 0 | | 0 | | 0 | | ns |
| $t_{RCH}$ ⑤ | | $\overline{\text{RCLK}}$ HIGH | 600 | | 600 | | 600 | | ns |
| $t_{RCL}$ ⑥ | | $\overline{\text{RCLK}}$ LOW | 4.0 | | 4.0 | | 4.0 | | μs |

Notes: 1. CCLK and DOUT timing are the same as for slave mode.
2. At power up, $V_{CC}$ must rise from 2.0 V to $V_{CC}$ minimum in less than 10 ms.

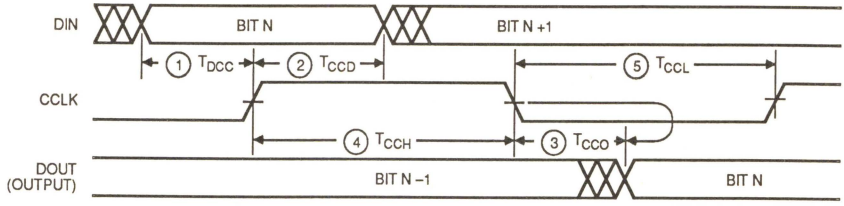



5

## Switching Characteristics — Programming — Slave Mode

| SYMBOL | DESCRIPTION | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{CCO}$ ③ | CCLK to DOUT | | 65 | | 65 | | 65 | ns |
| $t_{DCC}$ ① | CCLK DIN setup | 0 | | 0 | | 0 | | ns |
| $t_{CCD}$ ② | CCLK DIN hold | 40 | | | 40 | | 40 | ns |
| $t_{CCH}$ ④ | CCLK HIGH time | 0.25 | | 0.25 | | 0.25 | | μs |
| $t_{CCL}$ ⑤ | CCLK LOW time | 0.25 | 5.0 | 0.25 | 5.0 | 0.25 | 5.0 | μs |
| $F_{CC}$ | CCLK frequency | | 2 | | 2 | | 2 | MHz |

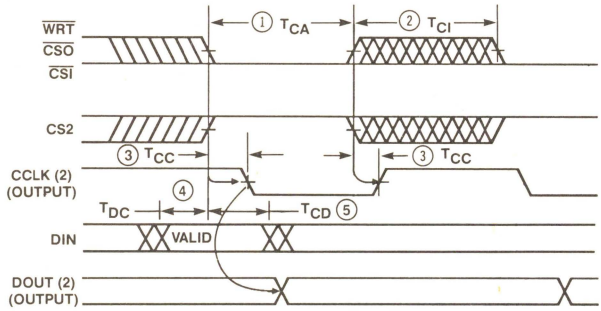
Note: Configuration must be delayed at least 40 ms after $V_{CC}$ minimum.



## Switching Characteristics — Programming — Peripheral Mode

| SYMBOL | DESCRIPTION | | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{CA}$ ① | Controls[1] ($\overline{CS0}$, $\overline{CS1}$, CS2, $\overline{WRT}$) | Active (last active input to first inactive) | 0.25 | 5.0 | 0.25 | 5.0 | 0.25 | 5.0 | μs |
| $t_{CI}$ ② | | Inactive (first inactive input to last active) | 0.25 | | 0.25 | | 0.25 | | μs |
| $t_{CCC}$ ③ | | CCLK[2] | | 75 | | 75 | | 75 | ns |
| $t_{DC}$ ④ | | DIN setup | 35 | | 35 | | 35 | | ns |
| $t_{CD}$ ⑤ | | DIN hold | 5 | | 5 | | 5 | | ns |

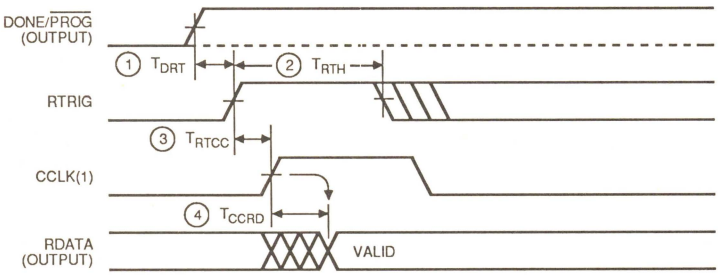
Notes: 1. Peripheral mode timing determined from last control signal of the logical AND of ($\overline{CS0}$, $\overline{CS1}$, CS2, $\overline{WRT}$) to transition to active or inactive state.
2. CCLK and DOUT timing are the same as for slave mode.
3. Configuration must be delayed at least 40 ms after $V_{CC}$ minimum.



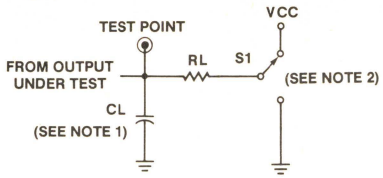

## Switching Characteristics — Program Readback

| SYMBOL | DESCRIPTION | | -33 | | -50 | | -70 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{DRT}$ ① | RTRIG | PROG setup | 300 | | 300 | | 300 | | ns |
| $t_{RTH}$ ② | | RTRIG HIGH | 250 | | 250 | | 250 | | ns |
| $t_{RTCC}$ ③ | CCLK | RTRIG setup | 100 | | 100 | | 100 | | ns |
| $t_{CCRD}$ ④ | | RDATA delay | | 100 | | 100 | | 100 | ns |

Notes: 1. CCLK and DOUT timing are the same as for slave mode.
2. DONE/PROG output/input must be HIGH (device programmed) prior to a positive transition of RTRIG (M0).



## Switching Test Load




Note: CL includes probe and jig capacitance.

CL = 50 pF
RL = 1K

## Design Aids

Designing with the Logic Cell Array is similar to using conventional MSI elements or gate array macrocells. The first step is to partition the desired logic design into Logic Blocks and I/O blocks, usually based on shared input variables or efficient use of flip-flop and combinatorial logic. Following a plan for placement of the blocks, the design information may be entered using the interactive Graphic Design Editor. The design information includes both the functional specifications for each block and a definition of the interconnection networks. A macrocell library provides a simplified entry of commonly-used logic functions. As an alternative to interactive block placement and configuration, a schematic may be created using elements from the macrocell library. Automatic placement and routing is available for either method of design entry. After routing the interconnections, various checking stages and processing of that data are performed to ensure that the design is correct. Design changes may be implemented in minutes. The design file is used to generate the programming data which can be downloaded directly into an LCA in the target system and operated. The program information may be used to program PROM, EPROM or ROM devices, or stored in some other media as needed by the final system.

Design verification may be accomplished by using the XLINX XACTOR™ In-Circuit Emulation System directly in the target system and/or the P-Silos™ logic simulator.

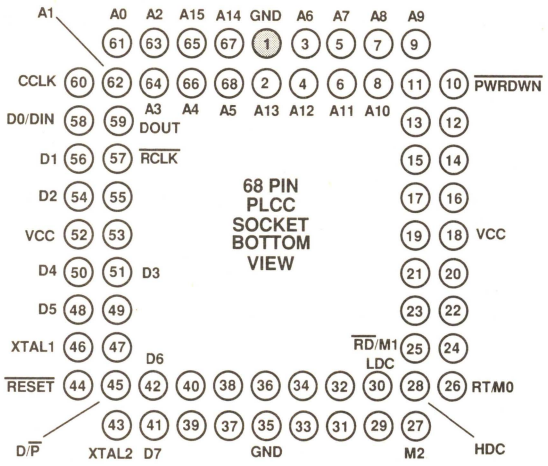
## Recommended Sockets

The following sockets, with matching hole patterns, are available for PLCC devices.

| | DESCRIPTION | VENDOR | PART NUMBER |
|---|---|---|---|
| 68-pin | PCB solder tail, tin plate | AMP | 821574-1 |
| | Surface mount, tin plate | AMP | 821542-1 |
| | PCB solder tail, tin plate | Burndy* | QILE68P-410T |
| | PCB solder tail, tin plate | Midland-Ross* | 709-2000-068-4-1-1 |
| | PCB solder tail, tin plate | Methode* | 213-068-001 |
| | Surface mount, tin plate | Methode* | 213-068-002 |
| 84-pin | PCB solder tail, tin plate | AMP | 821573-1 |
| | Surface mount, tin plate | AMP | 821546-1 |
| | PCB solder tail, tin plate | Burndy* | QILE84P-410T |
| | PCB solder tail, tin plate | Midland-Ross* | 709-2000-084-4-1-1 |
| | PCB solder tail, tin plate | Methode* | 213-084-001 |
| | Surface mount, tin plate | Methode* | 213-084-002 |

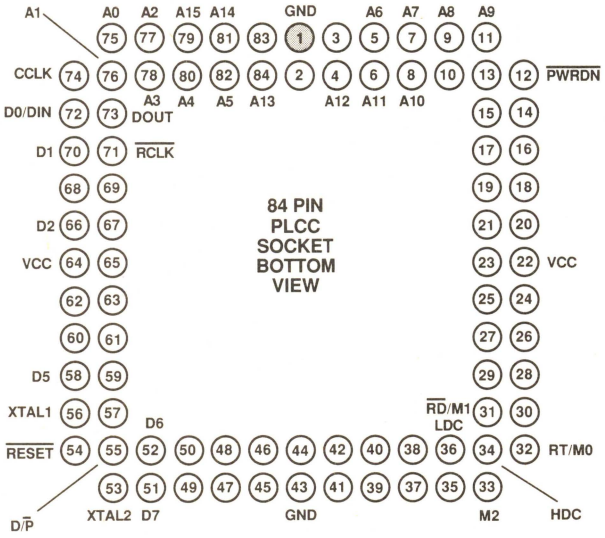
* Sockets will plug into pin-grid array (PGA) wire-wrap sockets for breadboard use.

M2064/18 PLCC SOCKET PIN ASSIGNMENT
WIRING REFERENCE. BOTTOM VIEW.




FOR EASE OF WIRING, AND PIN IDENTIFICATION, THE BOTTOM VIEW OF THE PLCC IS SHOWN ALONG WITH KEY PIN ASSIGNMENTS, SUCH AS ADDRESS, DATA, MODE, POWER AND CRYSTAL OSCILLATIOR INPUTS.

M2018 PLCC SOCKET PIN ASSIGNMENT
WIRING REFERENCE. BOTTOM VIEW.




FOR EASE OF WIRING, AND PIN IDENTIFICATION, THE BOTTOM VIEW OF THE PLCC IS SHOWN ALONG WITH KEY PIN ASSIGNMENTS, SUCH AS ADDRESS, DATA, MODE, POWER AND CRYSTAL OSCILLATIOR INPUTS.

5

# Military CMOS Programmable Gate Array Logic Cell™ Array M2064/M2018

**Conforms to MIL-STD-883, Class B***

## Features

CMOS
- Low power
- TTL or CMOS input threshold levels

PROGRAMABLE
- Programmable Logic functions
- Programmable I/O blocks
- Programmable interconnects

STATIC RAM BASED
- Reprogrammable
- Reconfigurable

SECURITY
- User selectable Security Mode
- Verification feature

## Benefits

- Reduced power supply
- Higher board densities

- Complete user control of design
- Instant prototyping
- Replaces SSI and MSI devices

- Easy design modification
- Selectable configuration modes
- 100% testable

- Protects proprietary designs
- Eases design debug

## Ordering Information

M2018 - 50 M P 84/883, B

- PART NUMBER: M2064 (1200 Gates, 58 I/OBs); M2018 (1800 Gates, 74 I/OBs)
- SPEED GRADE: -20 (20 MHz Toggle Rate); -33 (33 MHz Toggle Rate); -50 (50 MHz Toggle Rate)
- TEMPERATURE RANGE: M = −55°C to +125°C
- PACKAGING: P = Pin Grid Array; L = Leadless Chip Carrier (68 pin only)
- NUMBER OF PINS: 68 (68 Pins); 84 (84 Pins)
- PROCESSING: 883,B = MIL-STD-883, Class B

## Description

The Military CMOS Logic Cell Array bridges the gap between Programmable Logic Devices (PLDs) and gate arrays. This high-density, low-power Logic Cell Array device provides designers with both the density benefits of gate arrays and the programmability benefits of user-configurable devices.

The flexible architecture of the LCA is similar to that of a gate array, with an interior matrix of programmable logic blocks called Configurable Logic Blocks (CLBs), a surrounding ring of programmable I/O blocks (IOBs) and programmable interconnects used to define the overall device structure. Unlike gate arrays, LCA functionality is user defined by loading the internal writable storage cells with the configuration data. The reprogrammability of the SRAM-based LCA allows instant design modification on the bench and on the board. Due to the SRAM-based architecture of the LCA, the radiation tolerance data for the M2064 is similar to industry SRAMs which display Total Dose levels from $10^4$ to $10^6$ rads (Si).

Applications for the LCA cover a wide spectrum of uses. With its high gate density and low-power CMOS technology, the LCA is an ideal low-cost gate array good for prototyping as well as production. The LCA's SRAM-based architecture allows it to be used in applications that take advantage of its reconfigurability. Ground systems (radar) can use the LCA as reconfigurable hardware replacing several devices and saving board space. The SRAM-based logic of the LCA allows for pattern security of sensitive designs when the device is removed from its board. When a different mode backup device is needed the LCA provides excellent redundancy. In short, the LCA offers the total ASIC solution.

## Silicon Menu

| MMI PART | ORGANIZATION | EQUIVALENT GATE COUNT | CONFIGURABLE LOGIC BLOCKS | USER I/Os | CONFIGURATION PROGRAM BITS | MAX STANDBY CURRENT (CMOS INPUTS) | MAX STANDBY CURRENT (TTL INPUTS) | PACKAGES | MAX TOGGLE RATE BETWEEN CLBs |
|---|---|---|---|---|---|---|---|---|---|
| M2064-20 | 8x8 | 1200 | 64 | 58 | 12040 | 5 mA | 10 mA | 68LCC, 68PGA | 20 MHz |
| M2064-33 | 8x8 | 1200 | 64 | 58 | 12040 | 5 mA | 10 mA | 68LCC, 68PGA | 33 MHz |
| M2064-50 | 8x8 | 1200 | 64 | 58 | 12040 | 5 mA | 10 mA | 68LCC, 68PGA | 50 MHz |

* Latest revision.

10331A
JANUARY 1988

## Software/Hardware Menu

| MMI PARTS | |
|---|---|
| LCA-MDS21 | LCA Development System |
| LCA-MSC21 | LCA Development System Annual Support Agreement |
| LCA-MDS22 | LCA Simulator (P-SILOS™) |
| LCA-MDS23 | LCA Automatic Placement and Routing Program |
| LCA-MDS24 | LCA In-Circuit Emulator |

| MMI PARTS | |
|---|---|
| LCA-MDS28 | LCA Universal Pod |
| LCA-MDS27XX | LCA Package Specific Pod Headers |
| LCA-MDS31 | LCA Futurenet® Interface |
| LCA-MDS33 | LCA Daisy Interface |
| LCA-MDS34 | LCA Mentor Interface |
| LCA-MDSXX | LCA/OrCAD™ Package |
| LCA-MEK01 | LCA Evaluation Kit |

## Absolute Maximum Ratings

Supply voltage, $V_{CC}$ ..... −0.5 V to 7.0 V
Power down, $V_{CC}$ ..... 2.0 V to 7.0 V
Input voltage range ..... −1.5 V to 5.5 V
Voltage applied to three-state output ..... −0.5 V to 5.5 V
Storage temperature ..... −65°C to +150°C
Terminal temperature, Leadless Chip Carrier package (Soldering, 10 seconds) ..... 240°C
Thermal resistance, junction to case, $\theta_{jcmax}$, and junction to ambient, $\theta_{jamax}$

| Package | $\theta_{jcmax}$ | $\theta_{jamax}$ (Still air) |
|---|---|---|
| (L) Leadless Chip Carrier | 1.5°C/W | 32°C/W |
| (P) Pin Grid Array | 3°C/W | 45°C/W |

Maximum power dissipation ..... See Table 2
Maximum junction temperature ..... 175°C
Maximum current density ..... Contact factory

## Operating Conditions

| SYMBOL | PARAMETER | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage relative to GND | 4.5 | | 5.5 | V |
| $V_{IHT}$ | High level input voltage—TTL configuration | 2.0 | | $V_{CC}$ | V |
| $V_{IHC}$ | High level input voltage—CMOS configuration | 0.7 $V_{CC}$ | | $V_{CC}$ | V |
| $V_{ILT}$ | Low level input voltage—TTL configuration | 0 | | 0.8 | V |
| $V_{ILC}$ | Low level input voltage—CMOS configuration | 0 | | 0.2 $V_{CC}$ | V |
| $I_{IT}$ | Input leakage current—TTL configuration | ±1 | | | μA |
| $I_{IC}$ | Input leakage current—CMOS configuration | ±1 | | | μA |
| $I_{OZ}$ | Three-state output off current ($V_{CC}$ = 5.5 V) | ±10 | | | μA |
| $t_{OP}$ | Operating free-air temperature | −55 | | +125 | °C |

## Electrical Characteristics Over Operating Conditions

**Conforms to MIL-STD-883 Group A Subgroups 1, 2 and 3**

| SYMBOL | PARAMETER | | TEST CONDITION | | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | High level output voltage | | $V_{CC}$ = MIN | $I_{OH}$ = −4.0 mA | 3.7 | | | V |
| $V_{OL}$ | Low level output voltage | | $V_{CC}$ = MIN | $I_{OL}$ = 4.0 mA | | | 0.4 | V |
| $I_{CCO}$ | Quiescent operating power supply current | CMOS inputs | $V_{CC}$ = 5.0 V | | | | 5 | mA |
| | | TTL inputs | $V_{CC}$ = 5.0 V | | | | 10 | mA |
| $I_{CCPD}$ | Power down supply current | | $V_{CC}$ = 2.0 V | | | | 0.5 | mA |

## Power On Timing

The LCAs contain on-chip reset timing logic for power-up operation. To insure proper master mode system operation, VCC must rise from 2.0 V to minimum specification level in 10 ms or less. For other modes, initiation of configuration must be delayed for 60 ms after VCC reaches the minimum specified level.

## Test Conditions

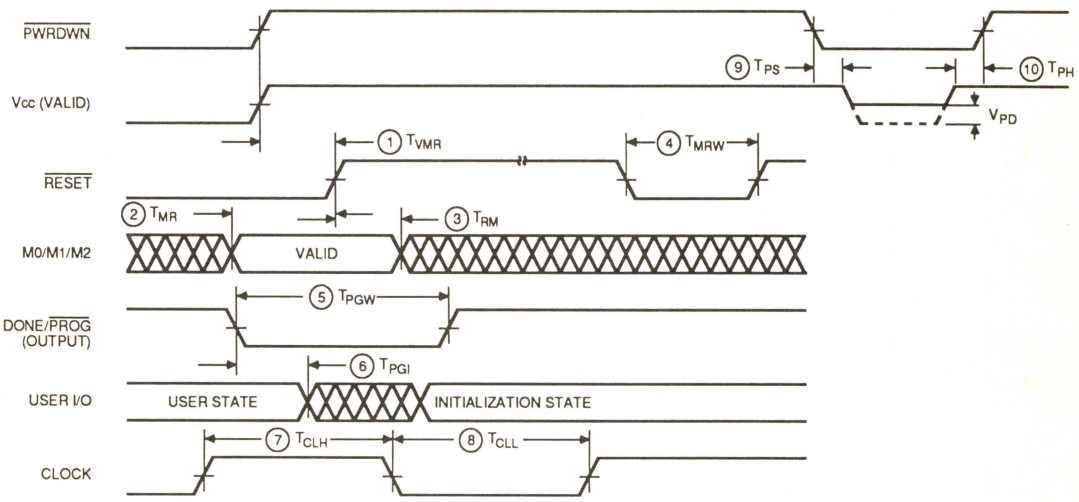
Outputs loaded with rated DC current and 50-pF capacitance to GND.

## Switching Characteristics — General

**Conforms to MIL-STD-883 Group A Subgroups 9, 10 and 11***

| SYMBOL | DESCRIPTION | | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{VMR}$ ① | $\overline{RESET}$[2] | $V_{CC}$ setup (2.0 V) | 250 | | 150 | | 150 | | ns |
| $t_{MR}$ ② | | M2, M1, M0 setup | 100 | | 60 | | 60 | | ns |
| $t_{RM}$ ③ | | M2, M1, M0 hold | 100 | | 60 | | 60 | | ns |
| $t_{MRW}$ ④ | | Width (LOW) | 250 | | 150 | | 150 | | ns |
| $t_{PGW}$ ⑤ | DONE/$\overline{PROG}$ | Program width (LOW) | 6 | | 6 | | 6 | | µs |
| $t_{PGI}$ ⑥ | | Initialization | | 7 | | 7 | | 7 | µs |
| $t_{CLH}$ ⑦ | CLOCK | Clock (HIGH) | 20 | | 12 | | 8 | | ns |
| $t_{CLL}$ ⑧ | | Clock (LOW) | 20 | | 12 | | 8 | | ns |
| $t_{PS}$ ⑨ | $\overline{PWR}$ $\overline{DWN}$ | Setup to $V_{CC}$ | 0 | | 0 | | 0 | | ns |
| $t_{PH}$ ⑩ | | Hold from $V_{CC}$ | 0 | | 0 | | 0 | | ns |
| $V_{PD}$ | | Power Down | 2.0 | | 2.0 | | 2.0 | | V |

Notes: 1. $V_{CC}$ must rise from 2.0 Volts to $V_{CC}$ minimum in lest than 10 ms for master mode.
2. $\overline{RESET}$ timing relative to power-on and valid mode lines (M0, M1, M2) is relevant only when $\overline{RESET}$ is used to delay configuration.
3. Minimum CLOCK widths for the auxiliary buffer are 1.25 times the $t_{CLH}$, $t_{CLL}$.
* Contact factory for test macros.



## Switching Characteristics — CLB

| SYMBOL | DESCRIPTION | | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{ILO}$ ① | Logic input to output | Combinatorial | | 35 | | 20 | | 15 | ns |
| $t_{ITO}$ ② | | Transparent latch | | 45 | | 25 | | 20 | ns |
| $t_{QLO}$ | | Additional for Q through F or G to out | | 30 | | 13 | | 8 | ns |
| $t_{CKO}$ ⑨ | K Clock | To output | | 35 | | 20 | | 15 | ns |
| $t_{ICK}$ ③ | | Logic-input setup | 22 | | 12 | | 8 | | ns |
| $t_{CKI}$ ④ | | Logic-input hold | 0 | | 0 | | 0 | | ns |
| $t_{CCO}$ ⑩ | C Clock | To output | | 45 | | 25 | | 19 | ns |
| $t_{ICC}$ ⑤ | | Logic-input setup | 18 | | 12 | | 9 | | ns |
| $t_{CCI}$ ⑥ | | Logic-input hold | 10 | | 6 | | 0 | | ns |
| $t_{CIO}$ ⑪ | Logic input to G Clock | To output | | 65 | | 37 | | 27 | ns |
| $t_{ICI}$ ⑦ | | Logic-input setup | 10 | | 6 | | 4 | | ns |
| $t_{CII}$ ⑧ | | Logic-input hold | 15 | | 9 | | 5 | | ns |
| $t_{RIO}$ ⑫ | Set/reset direct | Input A or D to out | | 45 | | 25 | | 22 | ns |
| $t_{RLO}$ ⑬ | | Through F or G to out | | 65 | | 37 | | 28 | ns |
| $t_{MRQ}$ | | Master Reset pin to out | | 60 | | 35 | | 25 | ns |
| $t_{RS}$ | | Separation of set/reset | 30 | | 17 | | 9 | | ns |
| $t_{RPW}$ | | Set/reset pulse-width | 20 | | 12 | | 9 | | ns |
| $F_{CLK}$ | Flip-flop toggle rate | Q through F to flip-flop | 20 | | 33 | | 50 | | MHz |
| $t_{CH}$ ⑭ | Clock | Clock HIGH | 20 | | 12 | | 8 | | ns |
| $t_{CL}$ ⑮ | | Clock LOW | 20 | | 12 | | 8 | | ns |

Note: All switching characteristics apply to all valid combinations of process, temperature and supply.

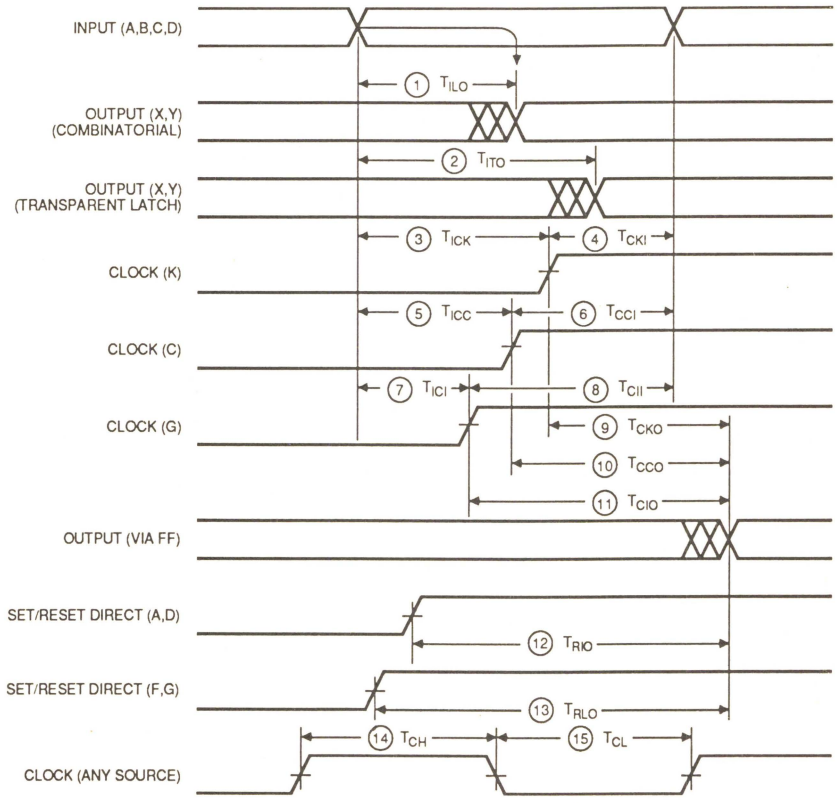
5

## Military Case Outlines*

| PACKAGE OUTLINE LETTER | CONFORMS TO MIL-M-38510 APPENDIX C CASE |
|---|---|
| L | C-7 |
| P | P-BC |

* Refer to MIL-M-38510, Appendix C for the appropriate package drawings.

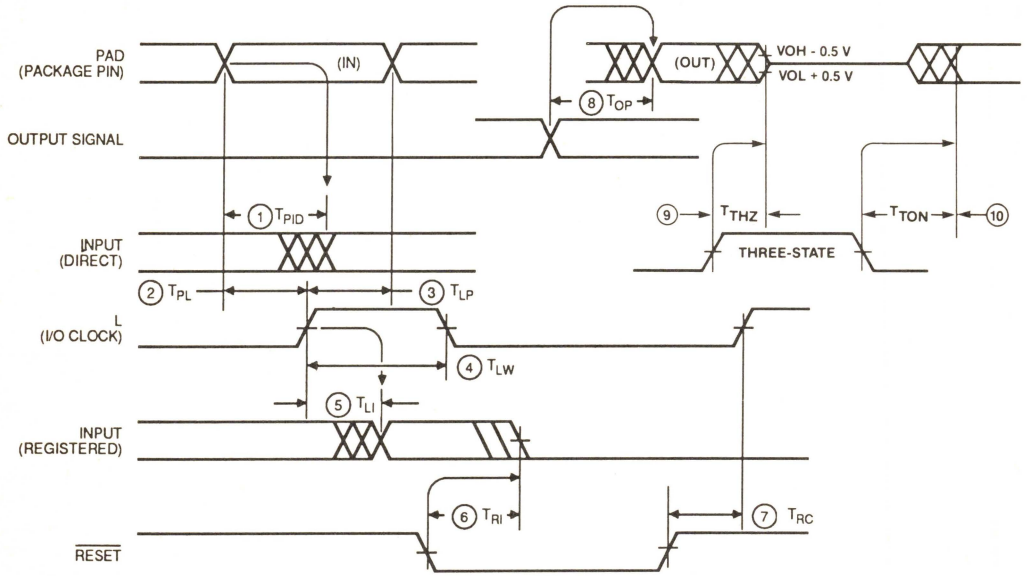
## Switching Characteristics CLB



## Switching Characteristics — IOB

| SYMBOL | DESCRIPTION | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{PID}$ ① | Pad (package pin) to input (direct) | | 20 | | 12 | | 8 | ns |
| $t_{LI}$ ⑤ | I/O Clock to input (storage) | | 30 | | 20 | | 15 | ns |
| $t_{PL}$ ② | /O Clock to pad-input setup | 20 | | 12 | | 8 | | ns |
| $t_{LP}$ ③ | I/O Clock to pad-input hold | 0 | | 0 | | 0 | | ns |
| $t_{LW}$ ④ | I/O Clock pulse width | 20 | | 12 | | 9 | | nsp |
| | I/O Clock frequency | | 20 | | 33 | | 50 | MHz |
| $t_{OP}$ ⑧ | Output to pad (output enabled) | | 25 | | 15 | | 12 | ns |
| $t_{THZ}$ ⑨ | Three-state to pad begin hi-Z | | 35 | | 25 | | 20 | ns |
| $t_{TON}$ ⑩ | Three-state to pad end hi-Z | | 40 | | 25 | | 20 | ns |
| $t_{RI}$ ⑥ | $\overline{\text{RESET}}$ to input (storage) | | 50 | | 40 | | 30 | ns |
| $t_{RC}$ ⑦ | $\overline{\text{RESET}}$ to input clock | | 35 | | 35 | | 25 | ns |

Note: Timing is measured at 0.5 $V_{CC}$ levels with 50-pF output load.

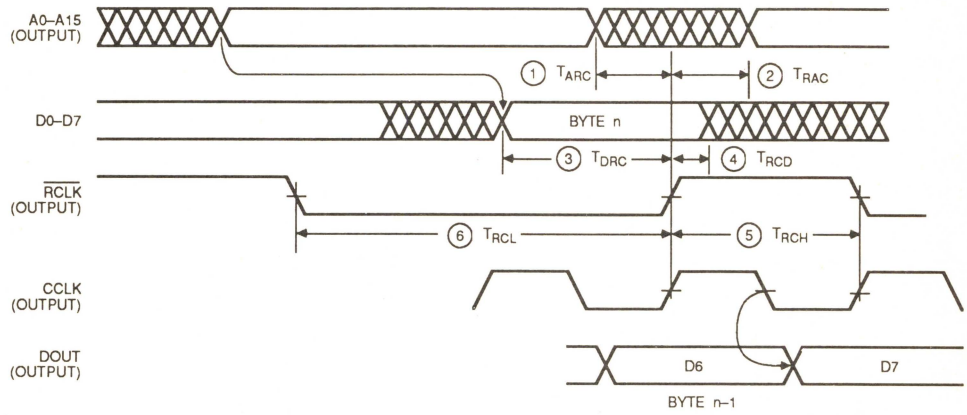



5

## Switching Characteristics — Programming — Master Mode

| SYMBOL | | DESCRIPTION | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{ARC}$ ① | RCLK | From address invalid | | 0 | | 0 | | 0 | ns |
| $t_{RAC}$ ② | | To address valid | | 300 | | 200 | | 200 | ns |
| $t_{DRC}$ ③ | | To data setup | 100 | | 60 | | 60 | | ns |
| $t_{RCD}$ ④ | | To data hold | 0 | | 0 | | 0 | | ns |
| $t_{RCH}$ ⑤ | | $\overline{\text{RCLK}}$ HIGH | 600 | | 600 | | 600 | | ns |
| $t_{RCL}$ ⑥ | | $\overline{\text{RCLK}}$ LOW | 4.0 | | 4.0 | | 4.0 | | μs |

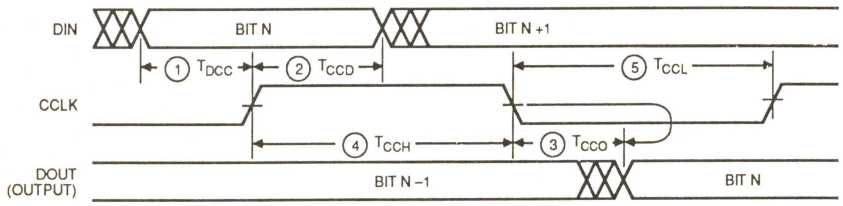
Notes: 1. CCLK and DOUT timing are the same as for slave mode.
2. At power up, $V_{CC}$ must rise from 2.0 V to $V_{CC}$ minimum in less than 10 ms.



## Switching Characteristics — Programming — Slave Mode

| SYMBOL | DESCRIPTION | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|
| | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{CCO}$ ③ | CCLK to DOUT | | 100 | | 65 | | 65 | ns |
| $t_{DCC}$ ① | CCLK DIN setup | 50 | | 25 | | 0 | | ns |
| $t_{CCD}$ ② | CCLK DIN hold | 75 | | | 40 | | 40 | ns |
| $t_{CCH}$ ④ | CCLK HIGH time | 0.50 | | 0.30 | | 0.25 | | μs |
| $t_{CCL}$ ⑤ | CCLK LOW time | 0.30 | 10.0 | 0.25 | 5.0 | 0.25 | 5.0 | μs |
| $F_{CC}$ | CCLK frequency | | 1 | | 2 | | 2 | MHz |

Note: Configuration must be delayed at least 40 ms after $V_{CC}$ minimum.

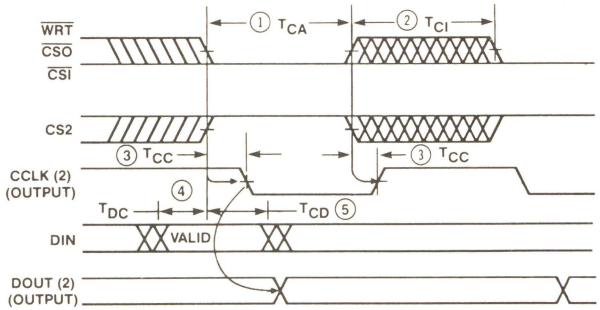



5

## Switching Characteristics — Programming — Peripheral Mode

| SYMBOL | DESCRIPTION | | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{CA}$ ① | Controls[1] ($\overline{CS0}$, $\overline{CS1}$, CS2, $\overline{WRT}$) | Active (last active input to first inactive) | 0.30 | 10.0 | 0.25 | 5.0 | 0.25 | 5.0 | µs |
| $t_{CI}$ ② | | Inactive (first inactive input to last active) | 0.25 | | 0.25 | | 0.25 | | µs |
| $t_{CCC}$ ③ | | CCLK[2] | | 100 | | 75 | | 75 | ns |
| $t_{DC}$ ④ | | DIN setup | 50 | | 35 | | 35 | | ns |
| $t_{CD}$ ⑤ | | DIN hold | 10 | | 5 | | 5 | | ns |

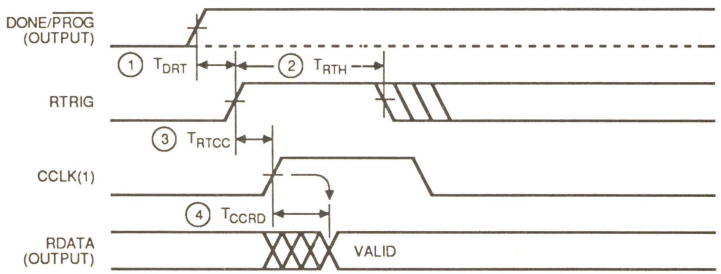
Notes: 1. Peripheral mode timing determined from last control signal of the logical AND of ($\overline{CS0}$, $\overline{CS1}$, CS2, $\overline{WRT}$) to transition to active or inactive state.
2. CCLK and DOUT timing are the same as for slave mode.
3. Configuration must be delayed at least 40 ms after $V_{CC}$ minimum.




## Switching Characteristics — Program Readback

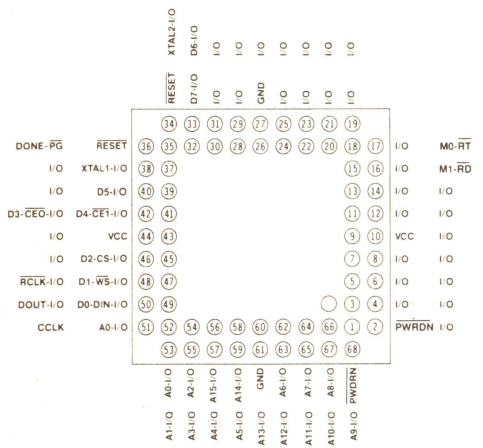
| SYMBOL | DESCRIPTION | | -20 | | -33 | | -50 | | UNIT |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN | MAX | MIN | MAX | MIN | MAX | |
| $t_{DRT}$ ① | RTRIG | PROG setup | 300 | | 300 | | 300 | | ns |
| $t_{RTH}$ ② | | RTRIG HIGH | 250 | | 250 | | 250 | | ns |
| $t_{RTCC}$ ③ | CCLK | RTRIG setup | 100 | | 100 | | 100 | | ns |
| $t_{CCRD}$ ④ | | RDATA delay | | 100 | | 100 | | 100 | ns |

Notes: 1. CCLK and DOUT timing are the same as for slave mode.
2. DONE/PROG output/input must be HIGH (device programmed) prior to a positive transition of RTRIG (M0).

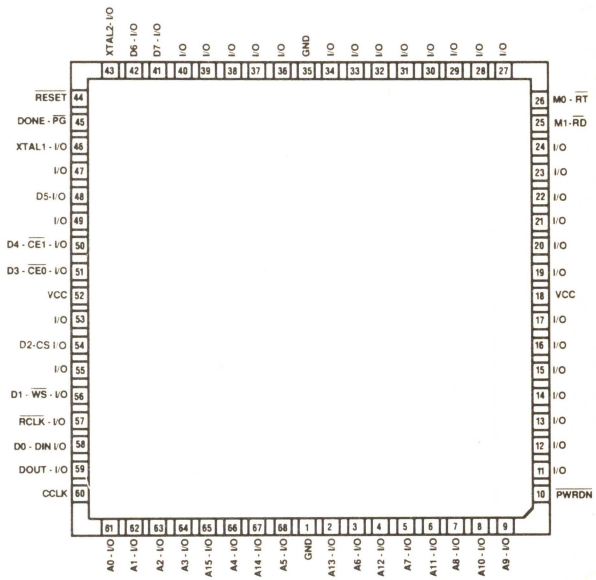


## Device Pinouts*

M2064
Logic Cell Array
68 Pin Grid Array
Top View

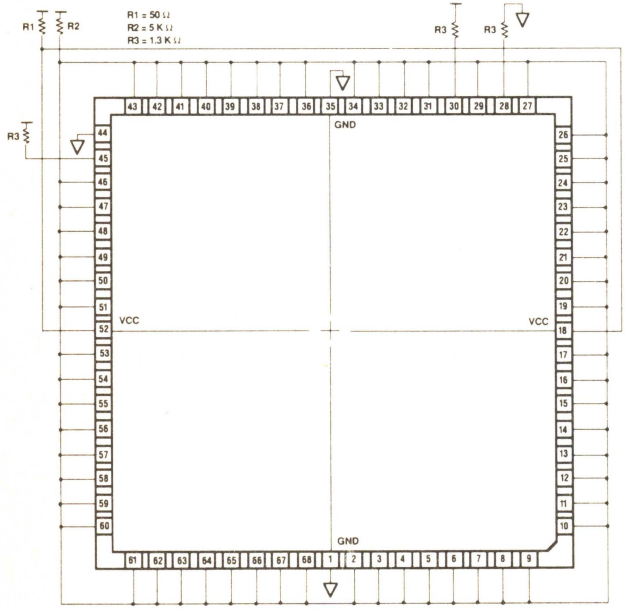



M2064
Logic Cell Array
68 Pin
Leadless Chip Carrier




## Burn-In Circuitry*

Condition C
Static Burn-In




Condition D
Dynamic Burn-In
Contact factory

5

## Configuration Pin Assignments*

| 68-PIN LCC | 68-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ | | | | USER OPERATION |
|---|---|---|---|---|---|---|
| | | SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | |
| 1 | B6 | GND | | | | |
| 2 | A6 | ≪HIGH≫ | | A13 (O) | | I/O |
| 3 | B5 | | | A6 (O) | | |
| 4 | A5 | | | A12 (O) | | |
| 5 | B4 | | | A7 (O) | | |
| 6 | A4 | | | A11 (O) | | |
| 7 | B3 | | | A8 (O) | | |
| 8 | A3 | | | A10 (O) | | |
| 9 | A2 | | | A9 (O) | | |
| 10 | B2 | $\overline{\text{PWRDWN}}$ (I) | | | | |
| 11 | B1 | ≪HIGH≫ | | | | I/O |
| 12 | C2 | | | | | |
| 13 | C1 | | | | | |
| 14 | D2 | | | | | |
| 15 | D1 | | | | | |
| 16 | E2 | | | | | |
| 17 | E1 | | | | | |
| 18 | F2 | VCC | | | | |
| 19 | F1 | ≪HIGH≫ | | | | I/O |
| 20 | G2 | | | | | |
| 21 | G1 | | | | | |
| 22 | H2 | | | | | |
| 23 | H1 | | | | | |
| 24 | J2 | | | | | |
| 25 | J1 | M1 (HIGH) | M1 (LOW) | M1 (HIGH) | M1 (LOW) | $\overline{\text{RDATA}}$ (O) |
| 26 | K1 | M0 (HIGH) | M0 (HIGH) | M0 (LOW) | M0 (LOW) | RTRIG (I) |
| 27 | K2 | M2 (HIGH) | | | | I/O |
| 28 | L2 | HDC (HIGH) | | | | |
| 29 | K3 | ≪HIGH≫ | | | | |
| 30 | L3 | LDC (LOW) | | | | |
| 31 | K4 | ≪HIGH≫ | | | | |
| 32 | L4 | | | | | |
| 33 | K5 | | | | | |
| 34 | L5 | | | | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 1. M2064 Pin Assignments**
**(continued on next page)**

## Configuration Pin Assignments*

| 68-PIN LCC | 68-PIN PGA | CONFIGURATION MODE: ≪M2: M1: M0≫ | | | | USER OPERATION |
|---|---|---|---|---|---|---|
| | | SLAVE <1:1:1> | PERIPHERAL <1:0:1> | MASTER-HIGH <1:1:0> | MASTER-LOW <1:0:0> | |
| 35 | K6 | GND | | | | |
| 36 | L6 | ≪HIGH≫ | | | | I/O |
| 37 | K7 | | | | | |
| 38 | L7 | | | | | |
| 39 | K8 | | | | | |
| 40 | L8 | | | | | |
| 41 | K9 | | | D7 (I) | | |
| 42 | L9 | | | D6 (I) | | |
| 43 | L10 | | | | | XTL2 or I/O |
| 44 | K10 | $\overline{\text{RESET}}$ (I) | | | | |
| 45 | K11 | DONE (O) | | | | PROG (I) |
| 46 | J10 | ≪HIGH≫ | | | | XTL1 or I/O |
| 47 | J11 | | | | | I/O |
| 48 | H10 | | | D5 (I) | | |
| 49 | H11 | | | | | |
| 50 | G10 | | $\overline{\text{CS0}}$ (I) | D4 (I) | | |
| 51 | G11 | | $\overline{\text{CS1}}$ (I) | D3 (I) | | |
| 52 | F10 | VCC | | | | |
| 53 | F11 | ≪HIGH≫ | | | | I/O |
| 54 | E10 | | $\overline{\text{CS2}}$ (I) | D2 (I) | | |
| 55 | E11 | | | | | |
| 56 | D10 | | $\overline{\text{WRT}}$ (I) | D1 (I) | | |
| 57 | D11 | | | $\overline{\text{RCLK}}$ | | |
| 58 | C10 | DIN (I) | | D0 (I) | | |
| 59 | C11 | DOUT (O) | | | | |
| 60 | B11 | CCLK (I) | CCLK (O) | | | CCLK (I) |
| 61 | B10 | ≪HIGH≫ | | A0 (O) | | I/O |
| 62 | A10 | | | A1 (O) | | |
| 63 | B9 | | | A2 (O) | | |
| 64 | A9 | | | A3 (O) | | |
| 65 | B8 | | | A15 (O) | | |
| 66 | A8 | | | A4 (O) | | |
| 67 | B7 | | | A14 (O) | | |
| 68 | A7 | | | A5 (O) | | |

≪HIGH≫ is high impedance with a 20 to 50-KΩ internal pull-up resistor during configuration

**Table 1. M2064 Pin Assignments**

* Contact factory for M2018 pinout, burn in circuitry and pin assignments.

## Pin Description

**$\overline{\text{PWRDWN}}$**

An active low power-down input stops all internal activity to minimize VCC power and puts all output buffers in a high-impedance state. Configuration is retained, however, internal storage elements are Reset. When the $\overline{\text{PWRDWN}}$ pin returns HIGH, the device returns to operation with the same sequence of reset, buffer enable and DONE, $\overline{\text{PROGRAM}}$ as at the completion of configuration.

**M0, RTRIG**

As Mode 0, this input and M1, M2 are sampled before the start of configuration to establish the configuration mode to be used.

As a read trigger, an input transition to a HIGH, after configuration is complete, will initiate a readback of configuration and storage element data. This operation may be limited to a single request, or be inhibited altogether, by selecting the appropriate readback option when generating the bit stream.

**M1, $\overline{\text{RDATA}}$**

As Mode 1, this input and M0, M2 are sampled before the start of configuration to establish the configuration mode to be used.

As an active-low read data; after configuration is complete, this pin is the output of the readback data.

**M2**

As Mode 2, this input and M0, M1 are sampled before the start of configuration to establish the configuration, mode to be used. After configuration, this pin becomes a user-programmable I/O.

**HDC**

High during configuration is held at a HIGH level by the LCA until after configuration. It is intended to be available as a control indication that configuration is not complete. After configuration, this pin is a user I/O.

**$\overline{\text{LDC}}$**

Low during configuration is held at a LOW level by the LCA until after configuration. It is intended to be available as a control indication that configuration is not completed. It is particularly useful in master mode as a LOW enable for an EPROM. After configuration, this pin is a user I/O. If used as a LOW EPROM enable, it should be programmed as a HIGH after configuration.

**$\overline{\text{RESET}}$**

This is an active-low input which has three functions. Prior to the start of configuration, a LOW input will delay the start of the configuration process. An internal circuit senses the application of power and begins a minimal time-out cycle on the order of 100 ms. When the time-out and $\overline{\text{RESET}}$ are complete, the levels of the "M" mode lines are sampled and configuration begins. If $\overline{\text{RESET}}$ is asserted during a configuration, the LCA is reinitialized and will restart the configuration at the termination of RESET. If $\overline{\text{RESET}}$ is asserted after configuration is complete, it will provide an asynchronous reset of all IOB and CLB storage elements of the LCA.

**DONE, $\overline{\text{PROG}}$**

The DONE open drain output is configurable with or without a pull-up resistor of about 3 KΩ. At the completion of configuration, the circuitry of the LCA becomes active in a synchronous order and one configuration clock cycle later DONE is asserted. Once configuration is done, a HIGH-to-LOW transition of this program pin will cause an initialization of the LCA and start a reconfiguration if that mode is selected in the current configuration.

**XTL1**

This user I/O pin may be configured to operate as the output of an amplifier usable with an external crystal and bias circuitry to form an oscillator.

**XTL2**

This user I/O pin may be configured to operate as the input of an amplifier usable with an external crystal and bias circuitry to form an oscillator.

**CCLK**

During configuration, configuration clock is an output of an LCA in either master or peripheral mode. LCAs in slave mode use it as a clock input. During a readback operation, it is an input clock for the configuration data being output.

**DOUT**

This user I/O pin is used during configuration to output serial configuration data out for daisy-chained slaves' data in.

**DIN**

This user I/O pin is used as serial data in during slave or peripheral configuration. This pin is D0 in master configuration mode.

**$\overline{\text{CSO}}$, $\overline{\text{CS1}}$, CS2, $\overline{\text{WRT}}$**

These four inputs represent a set of signals, three active low and one active high, which are used in the peripheral mode to control configuration data entry. The assertion of all four generates a LOW CCLK and shifts DOUT data. The removal of any assertion clocks in the DIN data present and causes a HIGH CCLK. In master mode, these pins become part of the parallel configuration byte (D4, D3, D2, D1). After configuration is complete, they are user-programmed I/O.

**$\overline{\text{RCLK}}$**

During master mode configuration, this pin represents a read clock of an external memory device. After configuration is complete, this pin becomes a user-programmed I/O.

**D0-D7**

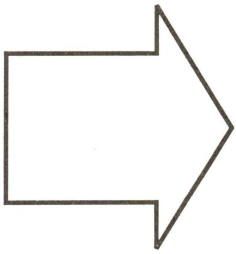
This set of eight pins represents the parallel configuration data byte for the master mode. After configuration is complete, they are user-programmed I/O.

**A0-A15**

This set of sixteen pins presents an address output for an external configuration memory during master mode. After configuration is complete, they are user-programmed I/O. A12 through A15 are not available in packages with less than sixty-eight pins.

**I/O**

A pin which may be programmed by the user to be input and/or output following configuration. Some of these pins present a high-impedance pull-up or perform other functions before configuration is complete.



## Data Sheets


TTL/CMOS PAL Devices

TTL/CMOS AmPAL Devices

PROSE/PLS Sequencers

FPC/PEG Sequencers

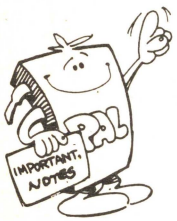
ECL PAL Devices

HAL/ZHAL Devices

Military PAL Devices

Logic Cell Array

Electrical Definitions


IMPORTANT
NOTES

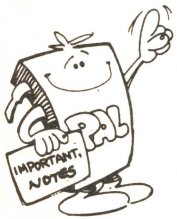
# Electrical Characteristic Definitions

| MMI | AMD | PARAMETER NAME | PARAMETER DEFINITION |
|---|---|---|---|
| **TIMING** | | | |
| $t_{ar}$ | $t_{ARO}$ | Asynchronous Preset Recovery Time | The minimum time after the asynchronous preset becomes inactive to the next input clock triggering edge. |
| $t_{aw}$ | $t_{AW}$ | Asynchronous Preset Width | The minimum pulse width required for the asynchronous preset signal. |
| $t_h$ | $t_H$ | Hold Time | The minimum time a valid data level is held after clock triggering edge. |
| $t_{hp}$ | N/A | Hold Time for Preload | The minimum delay time for data to remain stable after the preload signal becomes inactive. This only applies to TTL-level preload. |
| $t_{sr}$ | $t_{RO}$ | Synchronous Reset Recovery Time | The minimum time between the synchronous reset going inactive and the next input clock triggering edge. |
| $t_{su}$ | $t_S$ | Setup Time, Input or Feedback to Clock | The minimum time a valid data level of input or feedback is stable before the next clock triggering edge. |
| $t_{sup}$ | N/A | Data Setput Time for Preload | The minimum time for input data to be stable prior to the preload signal becoming inactive.This only applies to TTL-level preload. |
| $t_{wh}$ | $t_{CWH}$,$t_{WH}$ | Clock Width High | The minimum width of the clock high from rising edge to the next falling edge. In some cases, simultaneous minimum clock widths (both high and low) will exceed the minimum period of the device. |
| $t_{wl}$ | $t_{CWL}$,$t_{WL}$ | Clock Width Low | The minimum width of the clock low from falling edge to the next rising edge. In some cases, simultaneous minimum clock widths (both high and low) will exceed the minimum period of the device. |
| $t_{wp}$ | N/A | Preload Pulse Width | The minimum pulse width required to preload the registers. This only applies to TTL-level preload. |
| $t_{AP}$ | $t_{AP}$ | Asynchronous Preset to Output | The maximum time required to preset the register output after the preset signal is asserted. |
| $t_{ARO}$ | $t_{APO}$ | Asynchronous Reset to Output | The maximum time required to reset the register output after the reset signal is asserted. |
| $t_{CF}$ | N/A | Clock to Feedback | The maximum delay between the time the clock triggering edge is asserted and the signal appears on the feedback. |
| $t_{CLK}$ | $t_{CO1}$,$t_{CO}$ | Clock to Register Output or Feedback | The maximum time it takes to obtain a valid data level on the output pin after an input clock triggering edge is applied. |
| $t_{CR}$ | N/A | Input or Feedback to Registered Output from Combinatorial Configuration; Output Mux Select 1 to 0 | The minimum time from input or feedback to registered output as output mux selection changes from combinatorial to registered output (1 to 0). |
| $t_{EA}$ | $t_{EA}$ | Output Enable Time, Clock to Output | The minimum delay between when an input is asserted and the output switches from a high-impedance state to HIGH or LOW logic state. |

| MMI | AMD | PARAMETER NAME | PARAMETER DEFINITION |
|---|---|---|---|
| $t_{ER}$ | $t_{ER}$ | Output Disable Time, Input to Output | The minimum delay between when an input is asserted and the output switches from a HIGH or LOW logic state to a high-impedance state. |
| $t_F$ | N/A | Fall Time | The minimum time for a signal to fall from 80% to 20% of its stabilized high value. |
| $t_{PD}$ | $t_{PD}$ | Propagation Delay, Input or Feedback to Non-Reg. Output | The time for a signal to propagate from input or feedback to output. |
| $t_{PRH}$ | $t_{PRH}$ | Preset to Output | The minimum time for an output signal to be preset after the input preset is applied. |
| $t_{PXZ}$ | $t_{PXZ}$ | Output Disable Time, OE to Output | The minimum delay between when a dedicated enable signal is asserted and the output switches from a HIGH or LOW logic state to a high-impedance state. |
| $t_{PZX}$ | $t_{PZX}$ | Output Enable Time, OE to Output | The minimum delay between when a dedicated enable signal is asserted and the output switches from a high-impedance state to a HIGH or LOW logic state. |
| $t_R$ | N/A | Rise Time | The minimum time for a signal to rise from 20% to 80% of its stabilized high value. |
| $t_{RC}$ | N/A | Input or Feedback to Combinatorial Output from Registered Configuration; Output Mux Select 0 to 1 | The minimum time from input or feedback to combinatorial output as output mux selection changes from registered to combinatorial output (0 to 1) |
| N/A | $t_{CO2}$ | Clock to Register Feedback through Array to Combinatorial Output | The minimum delay between the clock triggering edge, signal appearing on the feedback, passing through the array and the data appearing on the combinatorial output. |
| N/A | $t_P$,$t_{P1}$ | Clock Period, External, $t_{su} + t_{CLK}$ | The minimum clock period when registered outputs are a function of both internal and external signals. It is specified from a rising (or falling) edge to the adjacent rising (or falling) edge. |
| N/A | $t_{P2}$ | Clock Period, Internal, $t_{su} + t_{CF}$ | The minimum internal delay from the register feedback, passing through the combinatorial logic, to register output. |
| **VOLTAGE** | | | |
| $V_{CC}$ | $V_{CC}$ | Supply Voltage, Positive Potential | The voltage required across supply and ground terminals of a TTL or CMOS integrated circuit. |
| $V_{EE}$ | $V_{EE}$ | Supply Voltage, Negative Potential | The voltage required across supply and ground terminals of an ECL integrated circuit. |
| $V_{IC}$ | $V_I$ | Input Clamp Voltage | The maximum input clamp voltage limit on every input pin. |
| $V_{IH}$ | $V_{IH}$ | High-Level Input Voltage | The minimum high-level input voltage that is guaranteed to represent a high logic level. |
| $V_{IL}$ | $V_{IL}$ | Low-Level Input Voltage | The maximum low-level input voltage that is guaranteed to represent a low logic level. |
| $V_{OH}$ | $V_{OH}$ | High-Level Output Voltage | The minimum high logic level guaranteed for all outputs. |
| $V_{OL}$ | $V_{OL}$ | Low-Level Output Voltage | The maximum low logic level guaranteed for all outputs. |

| MMI | AMD | PARAMETER NAME | PARAMETER DEFINTION |
|---|---|---|---|
| **CURRENT** | | | |
| $I_{CC}$ | $I_{CC}$ | Supply Current, Corresponding to $V_{CC}$ | The maximum current into the $V_{CC}$ terminal of a TTL or CMOS integrated circuit. |
| $I_{EE}$ | $I_{EE}$ | Supply Current, Corresponding to $V_{EE}$ | The maximum current into the $V_{EE}$ terminal of an ECL integrated circuit. |
| $I_I$ | $I_I$ | Input Current with Maximum Input Voltage | The maximum current into an input pin when the input voltage is applied to the input pin. |
| $I_{IH}$ | $I_{IH}$ | High-Level Input Current | The maximum current into an input pin when a logic-high level is applied to the input pin. |
| $I_{IL}$ | $I_{IL}$ | Low-Level Input Current | The maximum current into an input pin when a logic-low level is applied to the input pin. |
| $I_{OH}$ | $I_{OH}$ | High-Level Output Current | The maximum current into an output pin to guarantee an output logic-high level. |
| $I_{OL}$ | $I_{OL}$ | Low-Level Output Current | The maximum current into an output pin to guarantee an output logic-low level. |
| $I_{OS}$ | $I_{SC}$ | Output Short-Circuit Current | The current into an output when that output is short-circuited to ground. |
| $I_{OZH}$ | $I_{OZH}$ | High-Level Leakage Current | The maximum current into a high-impedance state output pin when a high logic level is applied to the output pin. |
| $I_{OZL}$ | $I_{OZL}$ | Low-Level Leakage Current | The maximum current into a high-impedance state output pin when a low logic level is applied to the output pin. |
| **MISCELLANEOUS** | | | |
| $C_{IN}$ | $C_{IN}$ | Input Capacitance | The input pin capacitance at a specified voltage and frequency. |
| $C_{OUT}$ | $C_{OUT}$ | Output Capacitance | The output or I/O pin capacitance at a specified voltage and frequency. |
| $T_A$ | $T_A$ | Operating Free Air Temperature | The ambient homogeneous temperature of the environment during operation. |
| $T_C$ | $T_C$ | Operating Case Temperature | The maximum chassis temperature during operation. |
| $f_{MAX,}$ External | $f_{1MAX}$ | Maximum External Frequency, $1/t_{P1}$ | The $f_{MAX, External}$ is the maximum clocking frequency with an external feedback. It is the reciprocal of the clock period $t_{P1}$ or $(t_{su} + t_{CLK})$. |
| $f_{MAX,}$ Internal | $f_{2MAX}$ | Maximum Internal Frequency, $1/t_{P2}$ | The $f_{MAX, Internal}$ is the maximum clocking frequency with an internal feedback. It is the reciprocal of the clock period $t_{P2}$ or $(t_{su} + t_{cF})$. |
| $f_{MAX,}$ without Feedback | N/A | Maximum Frequency without Feedback | The $f_{MAX, without\ Feedback}$ is the maximum clocking frequency with no feedback. It is the reciprocal of the sum of the data setup time ($t_{su}$) and the data hold time ($t_h$). |

5


IMPORTANT
NOTES

## PAL Device Handbook


Introduction 1

Applications 2

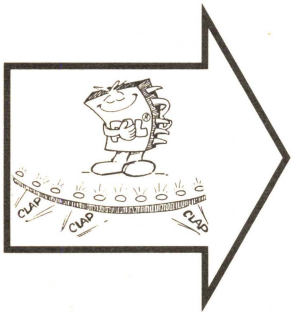

## PAL Device Data Book


Programming and Quality 3

PALASM 2 Software User Documentation 4

Data Sheets 5

Appendices 6

**Logic Reference** ..... 6-1
Basic Logic Elements ..... 6-1
Basic Storage Elements ..... 6-8
Binary Numbers ..... 6-15

**Signal Polarity** ..... 6-19

**Glossary** ..... 6-24

**Competitive Cross-Reference** ..... 6-30

**Worldwide Application Support** ..... 6-40

**Sales Offices** ..... 6-41

# Logic Reference

## Introduction

Throughout this handbook we have assumed that you have a good working knowledge of logic. Unfortunately, there always comes a time when you are called on to remember something which can only be found in that logic textbook which you threw away two years ago.

This section is intended to provide a quick review and reference of the basic principles of digital logic. We will cover three general areas:

- Basic logic elements
- Basic storage elements
- Binary numbers

Throughout the text, we will use the notation which you would use when implementing a design with PALASM software. If you are unfamiliar with the syntax, you will probably find it easy to understand as you read; if you wish for a more detailed explanation of the symbols, please refer to the software documentation in this handbook.

As this is a logic reference only, we cannot take on lengthy discussions, nor can we train you in the basic principles of digital logic if you have not previously been trained. In such a case we must refer you to your favorite logic textbook.

## Basic Logic Elements

In this section, we will discuss the concepts surrounding combinatorial logic functions.

### The Three Basic Gates

There are three basic logic gates from which all other combinatorial logic functions can be generated. These functions are *NOT*, *AND*, and *OR*. A truth table indicating these functions is shown in Table 1. Since they can be used to generate any function, they are said to be *functionally complete*.

| A | B | /A | A*B | A+B |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |

**Table 1. Truth Table for the NOT, AND, and OR Functions**

The standard schematic symbols used to represent these gates are shown in Figure 1.

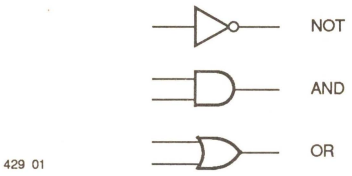



**Figure 1. Schematics Symbols for the Three Fundamental Gates**

The AND and NOT functions can be combined into the *NAND* function. This is equivalent to an AND gate followed by an inverter, as shown in Figure 2a. Likewise, the OR and NOT gates can be combined into the *NOR* function, as shown in Figure 2b. Each of these gates is functionally complete; any logic function can be expressed solely as a function of NAND or NOR gates.

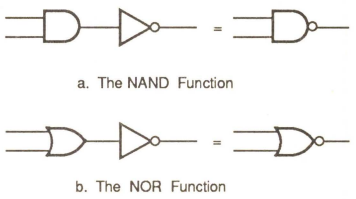



**Figure 2. The NAND and NOR Functions**

### Precedence of Operators

Logic functions may be created with any combination of the three basic functions. How those functions are expressed affects the evaluation of the function. The normal order of evaluation is:

NOT, AND, OR

Evaluation proceeds in order from left to right.

This order may be altered by inserting parentheses in the function. The contents of the parentheses will always be evaluated before the rest of the expression, from left to right.

Some example functions are evaluated in Table 2.

| A | B | C | D | A*B+/A* C+D | A*B+/A* (C+D) | A*(B+/A)* C+D | A*(B+/A)* (C+D) |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Table 2. Using Parentheses to Change the Order of Evaluation**

## Commutative, Associative, Distributive Laws

The AND and OR functions are commutative and associative. This means that the operands can appear in any order without affecting the evaluation of the function. This is illustrated in Tables 3 and 4.

| A | B | A*B | B*A | A+B | B+A |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

**Table 3. Commutativity**

| A | B | C | (A*B)*C | A*(B*C) | (A+B)+C | A+(B+C) |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Table 4. Associativity**

There are actually two distributive laws; one of them resembles standard algebra more than the other. These two laws state that:

$$A*(B + C) = (A*B) + (A*C)$$

$$A + (B*C) = (A + B)*(A + C)$$

## Duality

The two distributive laws give an example of the concept of *duality*. This principle states that:

- Any identity will also be true if the following substitutions are made:

  * for +
  \+ for *
  1 for 0
  0 for 1

Thus it is only necessary to prove the first of the distributive laws; the second one will then be true by duality. Note that duality is not required to prove the second law; it can also be proven by truth table or by logic manipulation.

## Manipulating Logic

Logic functions may be manipulated by the use of Boolean algebra. The logic functions may be expressed in one of the two canonical forms, or by using a simplified expression.

## Canonical Forms

There are two fundamental canonical forms: *sum-of-minterms* and *product-of-maxterms*. The former is by far the most widespread. These are special cases of what are more generally referred to as *sum-of-products* and *product-of-sums* forms. *Minterms* and *maxterms* are products and sums of the variables involved in a function. Each particular combination of non-inverted and inverted variables in a product or sum is given a minterm or maxterm number, as shown in Table 5. Within each minterm or maxterm, the individual variables are referred to as *literals*.

| MINTERM | NAME |
|---|---|
| /x*/y*/z | m0 |
| /x*/y* z | m1 |
| /x* y*/z | m2 |
| /x* y* z | m3 |
| x*/y*/z | m4 |
| x*/y* z | m5 |
| x* y*/z | m6 |
| x* y* z | m7 |

a. Table of Minterms for Three Variables

| MAXTERM | NAME |
|---|---|
| x + y + z | M0 |
| x + y + /z | M1 |
| x + /y + z | M2 |
| x + /y + /z | M3 |
| /x + y + z | M4 |
| /x + y + /z | M5 |
| /x + /y + z | M6 |
| /x + /y + /z | M7 |

b. Table of Maxterms for Three Variables

**Table 5. Minterms and Maxterms**

For the case of sum-of-minterms form, the expression for a function may be found by ORing the minterms which correspond to the 1's in the function's truth table. Likewise, the product-of-maxterms expression may be found by ANDing the maxterms which correspond to the 0's in the truth table. This is illustrated in Figure 3.

## Conversion Between Canonical Forms

It is a simple matter to convert between canonical forms. Given a truth table for a function F, there are four different representations that can be used:

- Sum-of-minterms form of F
- Product-of-maxterms form of F
- Sum-of-minterms form of /F
- Product-of-maxterms form of /F

One can convert back and forth between these representations by using the rules shown in Table 6.

| A | B | C | D | X | Y | MINTERM/ MAXTERM NUMBER |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 2 |
| 0 | 0 | 1 | 1 | 1 | 1 | 3 |
| 0 | 1 | 0 | 0 | 0 | 1 | 4 |
| 0 | 1 | 0 | 1 | 1 | 0 | 5 |
| 0 | 1 | 1 | 0 | 0 | 0 | 6 |
| 0 | 1 | 1 | 1 | 1 | 1 | 7 |
| 1 | 0 | 0 | 0 | 1 | 1 | 8 |
| 1 | 0 | 0 | 1 | 1 | 1 | 9 |
| 1 | 0 | 1 | 0 | 0 | 0 | 10 |
| . | . | . | . | . | . | . |
| . | . | . | . | . | . | . |
| . | . | . | . | . | . | . |
| 1 | 1 | 1 | 1 | 0 | 0 | 15 |

a. A Truth Table

```
X = m0+m2+m3+m5+m7+m8+m9
  = ∑m(0,2,3,5,7,8,9)

  = /A*/B*/C*/D        ;m0
  + /A*/B* C*/D        ;m2
  + /A*/B* C* D        ;m3
  + /A* B*/C* D        ;m5
  + /A* B* C* D        ;m7
  +  A*/B*/C*/D        ;m8
  +  A*/B*/C* D        ;m9

Y = m0+m1+m2+m3+m4+m7+m8+m9
  = ∑m(0,1,2,3,4,7,8,9)

  = /A*/B*/C*/D        ;m0
  + /A*/B*/C* D        ;m1
  + /A*/B* C*/D        ;m2
  + /A*/B* C* D        ;m3
  + /A* B*/C*/D        ;m4
  + /A* B* C* D        ;m7
  +  A*/B*/C*/D        ;m8
  +  A*/B*/C* D        ;m9
```

b. The Sum-Of-Minterms Expression

```
X = M1*M4*M6*M10*M11*M12*M13*M14*M15
  = ∏M(1,4,6,10,11,12,13,14,15)

  = ( A+ B+ C+/D)      ;M1
  * ( A+/B+ C+ D)      ;M4
  * ( A+/B+/C+ D)      ;M6
  * (/A+ B+/C+ D)      ;M10
  * (/A+ B+/C+/D)      ;M11
  * (/A+/B+ C+ D)      ;M12
  * (/A+/B+ C+/D)      ;M13
  * (/A+/B+/C+ D)      ;M14
  * (/A+/B+/C+/D)      ;M15

Y = M5*M6*M10*M11*M12*M13*M14*M15
  = ∏M(5,6,10,11,12,13,14,15)

  = ( A+/B+ C+/D)      ;M5
  * ( A+/B+/C+ D)      ;M6
  * (/A+ B+/C+ D)      ;M10
  * (/A+ B+/C+/D)      ;M11
  * (/A+/B+ C+ D)      ;M12
  * (/A+/B+ C+/D)      ;M13
  * (/A+/B+/C+ D)      ;M14
  * (/A+/B+/C+/D)      ;M15
```

c. The Product-of-Maxterms Expression

**Figure 3. Finding the Canonical Form From the Truth Table**

| GIVEN FORM | DESIRED FORM | | | |
|---|---|---|---|---|
| | MINTERM EXPANSION OF F | MAXTERM EXPANSION OF F | INVERTED MINTERM EXPANSION OF F | INVERTED MAXTERM EXPANSION OF F |
| Minterm expansion of F | – | Maxterm numbers are those numbers not in the Minterm list of F | List Minterms not present in F | Maxterm numbers are the same as Minterm numbers of F |
| Maxterm expansion of F | Minterm numbers are those numbers not on the Maxterm list of F | – | Minterm numbers are the same as Maxterm numbers of F | List Maxterms not present in F |

**Table 6. Conversion of Forms Table**

## Simplifying Logic

Canonical forms are convenient in that it is easy to derive and convert them. However, the representation is bulky, since all variables must appear in each sum or product. These expressions can be simplified by applying the basic laws and theorems of Boolean algebra.

There are four basic postulates, two of which are the commutative and distributive laws which were discussed above. From these postulates, it is possible to derive nine basic theorems. The postulates and theorems are listed in Table 7.

| | |
|---|---|
| **Postulate 1** | (A) X + FALSE = X<br>(B) X*TRUE = X |
| **Postulate 2** | (A) X + /X = TRUE<br>(B) X*/X = FALSE |
| **Postulate 3** | (A) X + Y = Y + X<br>(B) X*Y = Y*X |
| **Postulate 4** | (A) X*(Y + Z) = (X*Y) + (X*Z)<br>(B) X + (Y*Z) = (X + Y)*(X + Z) |
| **Theorem 1** | (A) X + X = X<br>(B) X*X = X |
| **Theorem 2** | (A) X + TRUE = TRUE<br>(B) X*FALSE = FALSE |
| **Theorem 3** | /(/X) = X |
| **Theorem 4** | (A) X + (Y + Z) = (X + Y) + Z<br>(B) X*(Y*Z) = (X*Y)*Z |
| **Theorem 5** | (A) /(X + Y) = /X*/Y<br>(B) /(X*Y) = /X + /Y |
| **Theorem 6** | (A) X + (X*Y) = X<br>(B) X*(X + Y) = X |
| **Theorem 7** | (A) (X*Y) + (X*/Y) = X<br>(B) (X + Y)*(X + /Y) = X |
| **Theorem 8** | (A) X + (/X*Y) = X + Y<br>(B) X*(/X + Y) = X*Y |
| **Theorem 9** | (A) (X*Y) + (/X*Z) + (Y*Z) = (X*Y) + (/X*Z)<br>(B) (X + Y)*(/X+ Z)*(Y+ Z) = (X + Y)*(/X + Z) |

**Table 7. Postulates and Theorems of Boolean Algebra**

Notice that each theorem and postulate (with the exception of theorem 3) has two forms. This is a result of the duality principle; once one form of a theorem is established, the dual representation follows immediately. Theorem 3 has no dual because it does not involve any of the elements that have duals (+, *, 1, or 0).

As the logic expression is simplified, it no longer contains minterms (or maxterms), since some of the minterms and literals are being eliminated. What was a sum-of-minterms (product of maxterms) representation is now simplified to a sum of products (product of sums).

## DeMorgan's Theorem

Once an expression has been simplified, it is no longer possible to invert the function by using Table 6. Inverting simplified logic requires DeMorgan's theorem:

```
/(X*Y) = /X + /Y
/(X + Y) = /X*/Y
```

This is theorem 5 in Table 7.

There is one shortcut which can be used. The effect of inversion can be accomplished by inverting all literals and then using the dual representation. For example, given the expression

```
/(A*/B + A*C + /A*B*D)
```

we can invert to obtain:

```
/A*B + /A*/C + A*/B*/D              ;step one,
                                     invert
                                     literals
(/A + B)*(/A + /C)*(A + /B + /D)    ;step two,
                                     take dual
```

This expression must still be simplified to obtain a sum-of-products representation, but this shortcut eliminates some of the early steps.

## Karnaugh Maps: Minimizing Logic

Simplifying by hand by using algebraic manipulation can be a tedious and error-prone procedure. When only a few variables are used (generally less than 5 or 6), Karnaugh maps (also called K-maps) provide a simpler graphical means of simplifying logic. K-maps not only allow for logic simplification, but for logic minimization, where an expression has a minimal number of product terms (or sum terms) and literals.

A Karnaugh map consists of a box which has one cell for each minterm. These cells are arranged so that only one literal is inverted when moving from one cell to an adjacent cell. The headings placed by each row and column indicate the polarities of the literals for that row or column. The literals themselves are indicated in the top left corner of the map. An example of a Karnaugh map for three variables is shown in Figure 4.

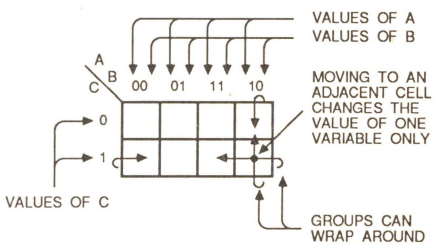



**Figure 4. A Karnaugh Map for Three Variables**

The truth table for a function is then transferred to the K-map by placing the 1's and 0's in the appropriate cells.

Since each cell differs from its neighbor only in the polarity of one of the literals, 1's in adjacent cells can be combined by theorem 7a, which says that

```
x*y + x*/y = x
```

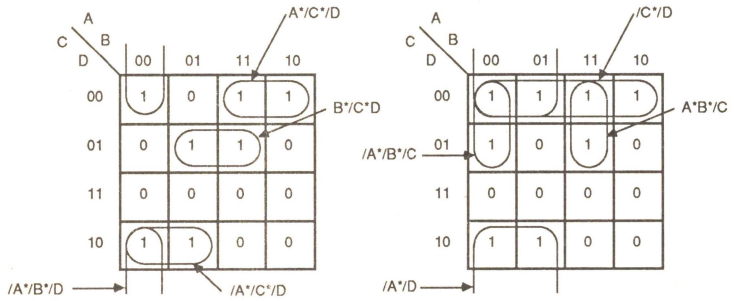
In this manner, two product terms are combined into one. This procedure can conceptually be repeated to allow groupings of two, four, eight, or any group of adjacent cells whose size is a power of two. A cell may appear in more than one group. Just enough groups are found to include all of the 1's. The groups should be as large as possible.

This process provides a minimal sum of products. The product-of-sums form can be obtained by grouping 0's instead of 1's and inverting the header for each cell.

The two functions from Figure 3 have been placed into K-maps in Figure 5. The groups are then used as individual product terms. When reading the product terms from the map, the only literals which will appear in the product term are the ones whose values are constant for each cell in the group. If that value is 1, then the non-inverted form of the literal is used. If the value is 0, then the inverted form of the literal is used.

For active-LOW functions, the same procedure is used, except that the 0's are grouped instead of the 1's. The active-LOW version of the functions from Figure 3 are derived in Figure 6.

Hand simplification and minimization is not needed as frequently today as in the past, since software is now available for handling these logic manipulations. PALASM software can perform logic simplification and minimization automatically.

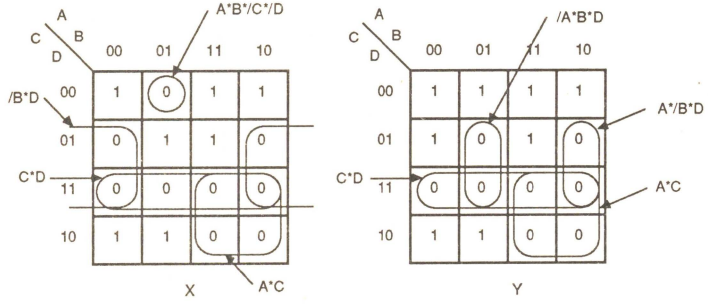



X = /A */B */D
+ A*/C*/D
+ B*/C*D
+ /A*C*/D

Y = /A*/D
+ /C*/D
+ /A*/B*/C
+ A*B*/C

429 04

**Figure 5. Using a K-map to Minimize the Functions in Figure 3**




/X = C*D
+ /B*D
+ A*C
+ /A*B*/C*/D

/Y = C*D
+ A*D
+ /A*B*D
+ A*/B*D

429 05

**Figure 6. Finding Inverse Functions**

## Comparison and Equivalence: the XOR and XNOR Gates

The Exclusive-OR (XOR) and Exclusive-NOR (XNOR) gates are two special gates which are relatively common. These gates have schematic symbols as shown in Figure 7a. They are actually compound gates, and can be generated by AND, OR, and NOT gates using the functions:

```
x :+: y = x*/y + /x*y     ;XOR gate

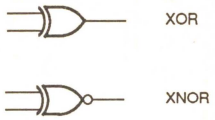
x :*: y = x*y + /x*/y     ;XNOR gate
```

The XOR and XNOR functions are actually inverses of each other; that is,

```
x :+: y = /(x :*: y)
```

Note that the XNOR operation is not explicitly supported by PALASM software. However, since it is the inverse of the XOR function, it is still possible to perform the equivalent logic.

The truth tables for these gates are shown in Figure 7b. Note that the XOR function is true if and only if the operands are different. For this reason, it is useful as a comparator. The XNOR function is true if and only if its operands are the same; therefore it is used as an equivalence indicator.




429 06

a. Schematic Symbols

| A | B | A:+:B | A:*:B |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

b. XOR and XNOR Truth Tables

**Figure 7. The Exclusive-OR and Exclusive-NOR Functions**

Some basic properties of the XOR and XNOR functions are listed in Table 8.

| XOR | XNOR |
|---|---|
| x :+: 0 = x<br>x :+: 1 = /x | x :*: 0 = /x<br>x :*: 1 = x |
| x :+: x = 0<br>x :+: /x = 1 | x :*: x = 0<br>x :*: /x = 1 |
| x :+: y = y :+: x<br>x :+: y :+: z = (x :+: y) :+: z<br>= x :+: (y :+: z) | x :*: y = y :*: x<br>x :*: y :*: z = (x :*: y) :*: z<br>= x :*: (y :*: z) |
| x :+: y = /x :+: /y | x :*: y = /x :*: /y |
| /(x :+: y) = /x :+: y<br>= x :+: /y<br>= x :*: y | /(x :*: y) = /x :*: y<br>= x :*: /y<br>= x :+: y |
| x :+: y = x*/y + /x*y | x :*: y = x*y + /x*/y |
| x :+: x*y = x*/y<br>x :+: /x*y = x + y | x :*: x*y = /x + y<br>x :*: /x*y = /x*/y |
| x*(y :+: z) = (x*y) :+: (x*z)<br>/x*(y :+: z) = (x + y) :+: (x + z) | x + (y :*: z) = (x + y) :*: (x + z)<br>/x + (y :*: z) = (x*y) :*: (x*z) |

**Table 8. Properties of the XOR and XNOR Functions**

When deriving equations from a Karnaugh map, XOR and XNOR functions can usually be identified by their characteristic pattern. Exactly what the operands are may or may not be obvious for more complicated functions. Some examples are shown in Figure 8.

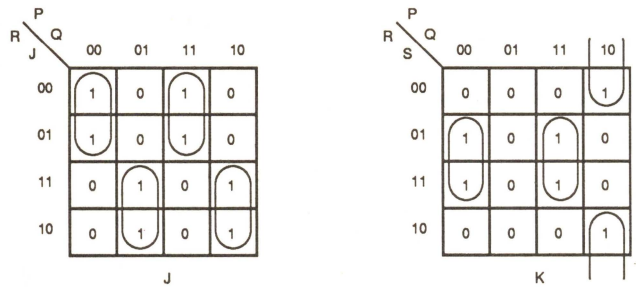
The XOR gate can be used as an "UNLESS" operator. In other words, the function

```
A = X :+: Y
```

can be interpreted as:

"A will have the same value as X UNLESS Y is true."

This can be helpful when trying to derive a logic equation for a function which can be described in words.




J = /P*/Q*/R
+ P*Q*/R
+ /P*Q*R
+ P*/Q*R
= ((/P*/Q) + (P*Q))*/R
+ ((/P*Q) + (P*/Q))*R
= (P :*: Q)*/R
+ (P :+: Q)*R
= /(P :+: Q)*/R
+ (P :+: Q)*R

= (P :+: Q) :*: R

K = /P*/Q*S
+ P*Q*S
+ P*/Q*/S
= ((/P*/Q) + (P*Q))*S
+ P*/Q*/S

= (P :*: Q)*S
+ P*/Q*/S

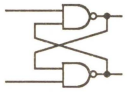
429 07

**Figure 8. Finding XOR and XNOR Functions in Karnaugh Maps**

6

## Basic Storage Elements

Storage elements provide circuits with the capability of remembering past conditions or events. The prototypical storage element is just a pair of cross-coupled NAND gates, as shown in Figure 9. These elements are normally called *flip-flops*.



429 08

**Figure 9. Basic Storage Element**

In general, there are two primary classes of flip-flops:

- *Unclocked* flip-flops, or *latches*
- *Clocked* flip-flops

Clocked flip-flops are sometimes referred to as *registers*, although technically speaking, a register is a bank of several flip-flops with a common clock signal.

Flip-flops can also be characterized by their control scheme. There are four types of flip-flops, each of which can be unclocked or clocked:

- S-R
- J-K
- D
- T

The discussion below will be divided between unclocked and clocked flip-flops. Each of the four flip-flop types will be treated for each section.

### Unclocked Flip-flops—Latches

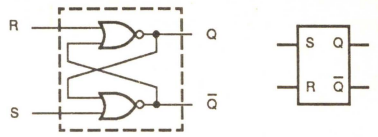
#### S-R Latches

An S-R latch can be built out of NOR gates as shown in Figure 10, and behaves according to the truth table in Table 9. 'S' stands for 'set' and 'R' stands for 'reset', as suggested by the truth table.

| S | R | Q+ |
|---|---|---|
| 0 | 0 | Q |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | Not allowed |

**Table 9. S-R Latch Truth Table**

Note that the latch actually has two outputs, which are complementary. These are referred to as Q and $\overline{Q}$. If both S and R are raised at the same time, then both Q and $\overline{Q}$ will be HIGH; although this is physically possible, it does not make sense if Q and $\overline{Q}$ are to be complementary signals. Thus this condition is not allowed.

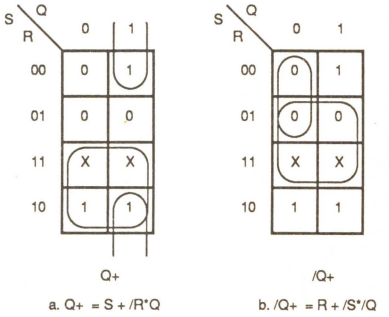



429 09

**Figure 10. An S-R Latch**

The transfer function for this latch can be derived with a Karnaugh map, as shown in Figure 11. By choosing either 1's or 0's, we can obtain two representations:

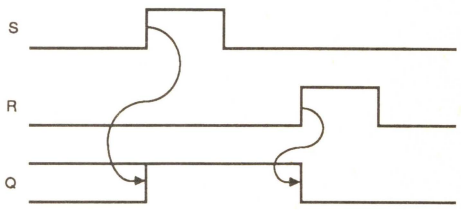
```
Q+ = S          /Q+ = R
   + /R*Q            + /S*/Q
```




429 10

**Figure 11. Karnaugh Maps for an S-R Latch**

Waveforms illustrating the operation of the S-R latch are shown in Figure 12.

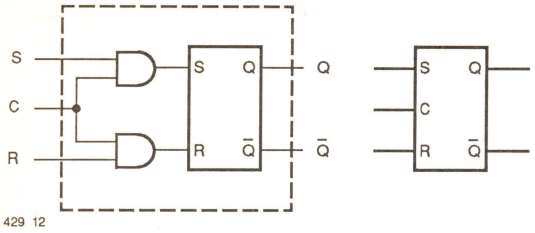



429 11

**Figure 12. S-R Latch Behavior**

There are some applications where it is desirable for the input data to be effective only when another signal—usually called a control signal—is active. The circuit of Figure 10 can be modified to give an S-R latch with a control input, as shown in Figure 13. The operation of this circuit is summarized in Table 10 and Figure 14.

The S-R latch is somewhat restrictive, since both inputs cannot be HI at the same time. The other latch types are based on the S-R latch, but have additional logic which removes the input restrictions.

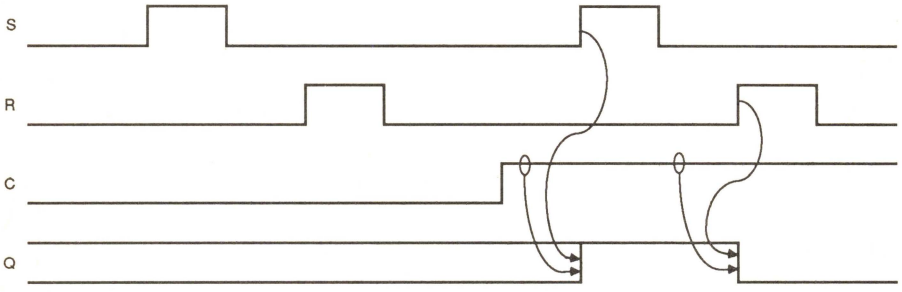



429 12

Figure 13. Adding a Control Input to an S-R Latch

| S | R | C | Q+ |
|---|---|---|---|
| X | X | 0 | Q |
| 0 | 0 | 1 | Q |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | Not allowed |

Table 10. Truth Table for an S-R Latch with a Control Input

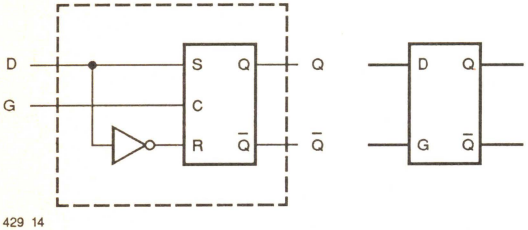



429 13

Figure 14. Behavior of an S-R Latch with a Control Input

### D-type Latches (Transparent Latches)

A single-input latch can be formed by adding some logic to the controlled S-R latch in Figure 13; this gives rise to the D-type latch in Figure 15. This latch is often called a *transparent* latch, since data on the input passes right through to the output as long as the control input is HIGH. If the control input is set LOW, then the latch holds whatever data was present when the control went LOW. With this type of latch, the control is usually called a *gate*.



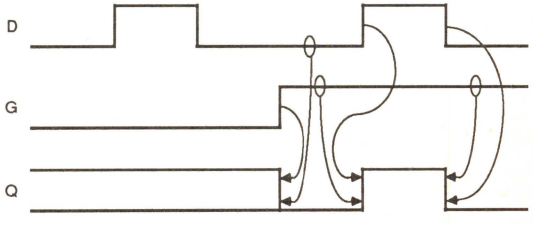

429 14

Figure 15. A D-type (Transparent) Latch

The behavior of the D-type latch is shown in Table 11 and Figure 16.

| D | G | Q+ |
|---|---|---|
| X | 0 | Q |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

Table 11. Truth Table for a D-type Latch




429 15

Figure 16. D-type (Transparent) Latch Behavior

The basic transfer function for a D-type latch can be derived from the Karnaugh map in Figure 17:

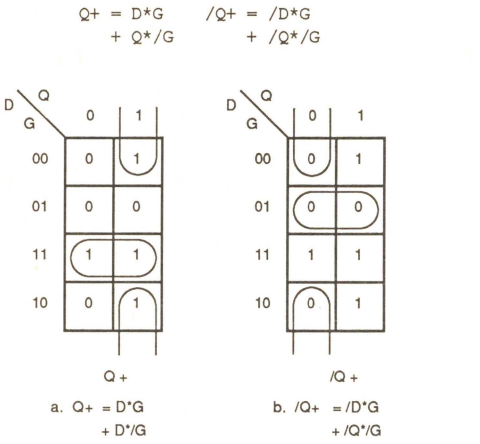



**Figure 17. Karnaugh Maps for a D-type Latch**

If realized exactly as the transfer function indicates, the result is actually a glitchy circuit. This is discussed in more detail on page 3-108 of the testability section. There a more practical transfer function is derived.

### J-K Latches

Another two-input latch can be derived from the S-R latch as shown in Figure 18. This is called a J-K latch, and operates in the same manner as an S-R latch, except that the condition where both inputs are HIGH is now allowed. The truth table is shown in Table 12; the waveforms are shown in Figure 19.

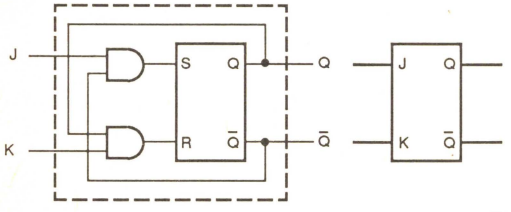



**Figure 18. A J-K Latch**

| J | K | Q+ |
|---|---|---|
| 0 | 0 | Q |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | $\overline{Q}$ |

**Table 12. Truth Table for a J-K Latch**

There are still some potential problems here for the case where J and K are both HIGH. If J and K are left HIGH for too long, the output may change more than one time; if left HIGH forever, the output will oscillate. Thus J and K should not be asserted for a time longer than the propagation delay of the latch. There are also potential race conditions if J and K are not asserted and removed at exactly the same time. If one of the inputs is raised slightly ahead of the other, it may give the output time to react, giving the wrong output once the second input is raised. The same problem can occur if one input is lowered slightly before the other. This is illustrated in Figure 20.

There are several ways to derive transfer functions for J-K latches. Two can be derived directly from Karnaugh maps, as shown in Figure 21; the others are not as obvious, and make use of the XOR gate described above. The basic transfer functions are listed in Table 13.

| | |
|---|---|
| Q+ = J*/Q + /K*Q | /Q+ = /J*/Q + K*Q |
| Q+ = Q :+: (J*/Q + K*Q) | /Q+ = /Q :+: (J*/Q + K*Q) |
| Q+ = /Q :+: (/J*/Q + /K*Q) | /Q+ = Q :+: (/J*/Q + /K*Q) |

**Table 13. Transfer Functions for a J-K Latch**

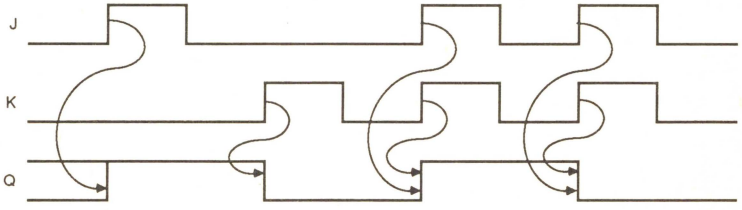
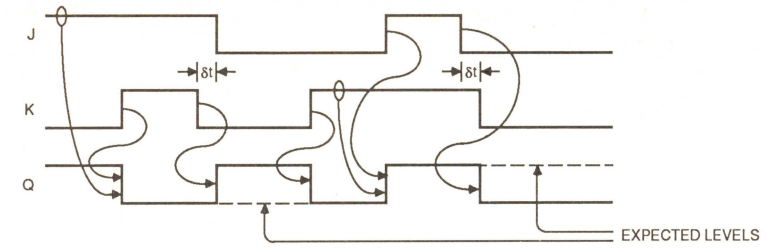
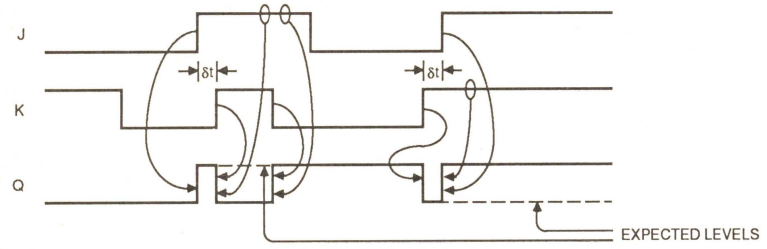


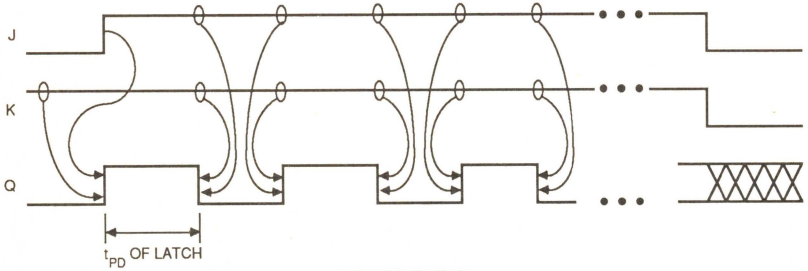

**Figure 19. Behavior of a J-K Latch**




a. Falling Edge Race Conditions.



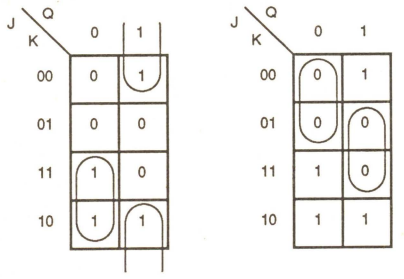

b. Rising Edge Race Conditions.




c. Possible Oscillation.

429 19

**Figure 20. Hazards Inherent in a J-K Latch**

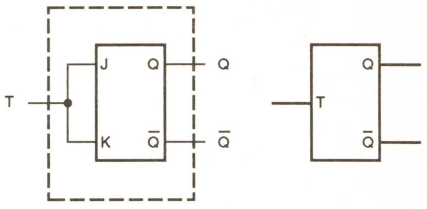



429 20

**Figure 21. Karnaugh Maps for a J-K Latch**

### T-type Latches

T-type latches are formed by connecting the J and K inputs of a J-K latch together to form a single input, as shown in Figure 22. This latch has two possible functions: hold the present state or invert the output, as summarized in Table 14. 'T' stands for 'trigger' or 'toggle', depending on who you talk to. That is, when T is HIGH, a change at the output is triggered; or, put another way, raising T causes the output to toggle.

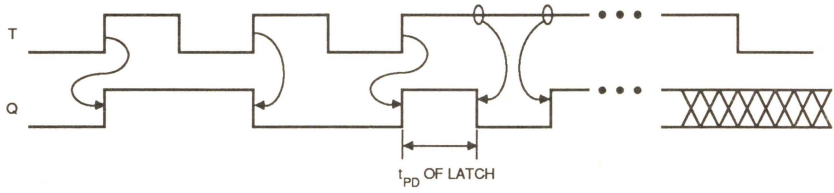



429 21

Figure 22. A T-type Latch

| T | Q+ |
|---|---|
| 0 | Q |
| 1 | /Q |

Table 14. The Truth Table for a T-type Latch



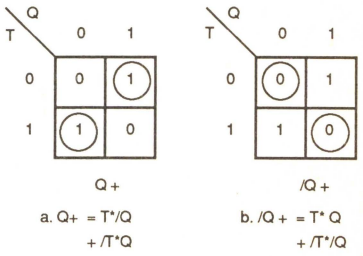

429 22

Figure 23. Behavior of a T-type Latch

This latch also has the problem that if T is left HIGH for too long, the output will oscillate. Since there is only one input, however, the race condition problems of the J-K latch have been eliminated. Unfortunately, this comes at the cost of initialization. There is now no way to get the output into a fixed state without knowing what the previous state was. Thus this device is not very useful without some kind of initialization circuit.

The general waveforms for a T-type latch are shown in Figure 23.

From the Karnaugh map in Figure 24, we can generate the following transfer functions:




429 23

Figure 24. Karnaugh Maps for a T-type Latch

```
Q+ = T*/Q                    /Q+ = T*Q
   + /T*Q                        + /T*/Q

Q+ = Q:+:T                   /Q+ = /Q :+: T

 Q+ =/Q:+:/T                 /Q+ = Q :+: /T
```

## Clocked Flip-flops

Latches can be modified by adding a *clock* input. The purpose of the clock is to delay any output changes until the clock signal changes. Whereas latch control inputs (such as the gate) are *level-sensitive*, clock inputs are generally *edge-sensitive* (or *edge-triggered*), meaning that output transitions can occur only when a clock transition is detected. A device is classified as positive edge-triggered or negative edge-triggered, depending on whether it responds to the rising or falling edge of the clock signal, respectively. The behavior of a clocked S-R flip-flop is illustrated in Figure 25.

The clock provides two basic advantages. It removes the hazards inherent in the J-K and T flip-flops, since all inputs will have settled by the time the clock edge arrives, and only one transition is possible for each clock edge. The clock also allows the design of synchronous systems, where all signals are coordinated with other signals. The entire system is then regulated by the clock.

The basic behavior of the four flip-flops types does not change with the addition of a clock; the output changes are merely made to wait for the clock edge. Thus the basic transfer equations for most of the flip-flops are the same. In the context of PALASM software, we can indicate the clocked nature of the flip-flops by using the "registered" assignment ':=' instead of '='.

### D-type Flip-flops

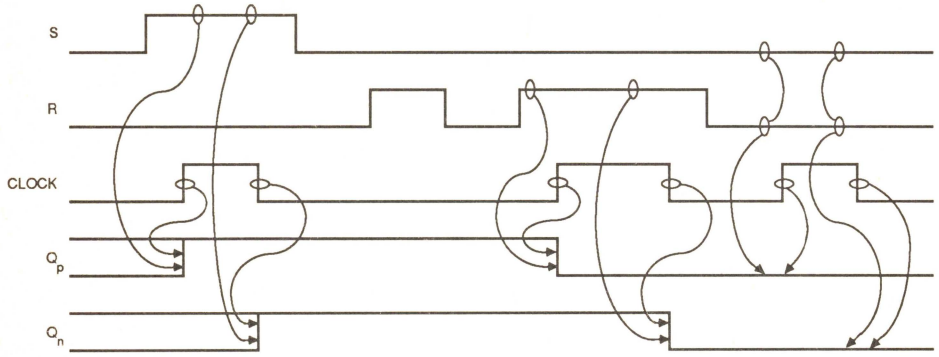
This is the only flip-flop type whose basic transfer characteristic changes, because the clock input replaces the gate input. Thus the transfer equations become

```
Q+ := D          /Q+ := /D
```

That is, whatever data appears on the input will be transferred to the output after the next clock edge. The input is not changed in any way.

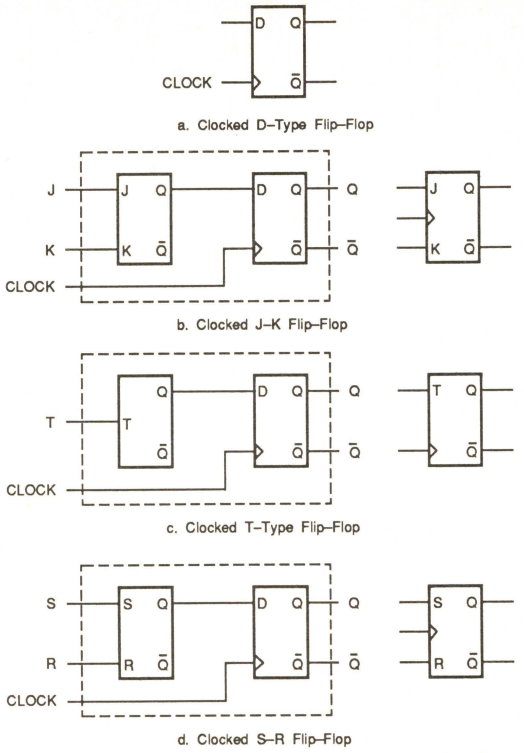
The simplicity of this flip-flop makes it the most widely used flip-flop. However, functions are sometimes more conveniently expressed using J-K flip-flops, or using T-type flip-flops. If we replace the D signal with the transfer function for one of the other flip-flop types, we can then emulate that flip-flop type in the D-type flip-flop. This is equivalent to taking a latch and placing a clocked D-type flip-flop after the latch output for synchronization. Figure 26 illustrates how each flip-flop can be emulated in a D-type flip-flop. The standard schematic symbols for the flip-flop types are also shown.

Table 15 summarizes the transfer functions for all of the flip-flop types. These functions can directly be used to emulate a particular flip-flop type in a D-type flip-flop. This can be particularly useful since D-type flip-flops are available in most registered PLDs.




429 24

**Figure 25. Behavior of a Clocked S-R Flip-flop for Positive (Qp) and Negative (Qn) Edge-triggered S-R Flip-flops**




429 25

**Figure 26. Clocked Flip-flops. All can be Emulated with a D-type Flip-flop.**

| | | |
|---|---|---|
| D-type | Q+ := D | /Q+ := /D |
| J-K-type | Q+ := J*/Q<br>+ /K*Q<br><br>Q+ := Q<br>:+: (J*/Q<br>+ K*Q)<br><br>Q+ := /Q<br>:+: (/J*/Q<br>+ /K*Q) | /Q+ := /J*/Q<br>+ K*Q<br><br>/Q+ := /Q<br>:+: (J*/Q<br>+ K*Q)<br><br>/Q+ := Q<br>:+: (/J*/Q<br>+ /K*Q) |
| T-type | Q+ := T*/Q<br>+ /T*Q<br><br>Q+ := Q :+: T<br><br>Q+ := /Q :+: /T | /Q+ := T*Q<br>+ /T*/Q<br><br>/Q+ := /Q :+: T<br><br>/Q+ := Q :+: /T |
| S-R-type | Q+ := S<br>+ /R*Q | /Q+ := R<br>+ /S*/Q |

**Table 15. Clocked Flip-flop Transfer Functions**

## Binary Numbers

The concept of a *number* is taken for granted by most people. And most people equate numbers in general with the *decimal* system, with which we are most familiar. However, there is nothing particularly special about the decimal system; the choice of system is actually rather arbitrary. History has chosen the decimal system for most humans.

For electronic systems, the *binary* system is more appropriate. It makes possible arithmetic and logical calculations that would be much more difficult—likely impractical—if implemented directly in a decimal system. Closely related to the binary system are the *octal* and *hexadecimal* systems, which will also be discussed here. Arithmetic is normally performed using binary numbers in a computer. Octal and hexadecimal representations are generally used as a way to "abbreviate" what might otherwise be lengthy binary numbers. This will be seen in when conversion is discussed below.

There are several terms which must be defined before proceeding further. A *number* is an abstract entity which is used to describe quantity. There are many ways of representing a number. Normally, the representation is designed around a *base*. The number is expressed as a sum of multiples of the powers of the base. The decimal system is a base-10 system, meaning that 10 is used as the base. The binary system is base-2; the octal system is base-8; and the hexadecimal system is base-16. The binary, octal, and hexadecimal systems are closely related because 8 and 16 are both powers of 2. When different bases are being used, a number will often be followed by its base in subscript, to indicate exactly what the base is. For example, the decimal number 25 would be written $25_{10}$ if its base were in doubt.

A number can thus be expressed in terms of some base x as follows:

$$a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x^1 + a_0 x^0 + a_{-1}x^{-1} + \ldots + a_{-m}x^{-m} \qquad (1)$$

The numbers $a_n \ldots a_{-m}$ are called *digits*. The value of each digit can range from 0 to x–1. Each digit is represented by a symbol, called a *numeral*. x numerals are required to represent a number in base x. The most familiar numerals are the symbols '0', '1',...'9'. There are ten of them, since they are used for the decimal system. For binary numbers, only '0' and '1' are used; for octal numbers, the numerals '0' through '7' are used. Hexadecimal numbers are more difficult, since sixteen numerals are required. Therefore the numerals '0' through '9' are used to represent the quantities $0_{10}$ through $9_{10}$; the letters A through F are used to represent the quantities $10_{10}$ through $15_{10}$.

The number expressed by equation 1 is normally represented as a string of digits:

$$a_n a_{n-1} \ldots a_1 a_0 . a_{-1} \ldots a_{-m}$$

The digits representing negative powers of the base are separated from those representing non-negative powers by a *point*. In the decimal system, this is referred to as a *decimal point*; in the binary system, it is referred to as a *binary point*.

There are two basic classes of manipulation which will be discussed: conversions between bases and arithmetic within a base.

### Converting Between Bases

Base-2 <—> Base-10

Converting a binary number to a decimal number is accomplished by using equation 1 directly.

Example:

Converting $110100.011_2$ to decimal:

$$\begin{aligned} Y &= 110100.011 \\ &= 1\cdot2^5 + 1\cdot2^4 + 0\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 0\cdot2^0 + 0\cdot2^{-1} + 1\cdot2^{-2} + 1\cdot2^{-3} \\ &= 32 + 16 + 4 + .25 + .125 \\ &= 52.375 \end{aligned}$$

When converting whole numbers from decimal to binary, the decimal number is repeatedly divided by 2. Integer division is used, so the quotients are "rounded down" to the next integer. The remainders form the digits of the number. The least significant digit is the first one calculated.

Example:

Converting $61_{10}$ to binary:

| | | |
|---|---|---|
| 61/2 = 30 | remainder = 1 | LSB |
| 30/2 = 15 | remainder = 0 | |
| 15/2 = 7 | remainder = 1 | |
| 7/2 = 3 | remainder = 1 | |
| 3/2 = 1 | remainder = 1 | |
| 1/2 = 0 | remainder = 1 | MSB |

$61_{10} = 111101_2$

When converting a decimal fraction into a binary fraction, the decimal number is multiplied by 2. This results in a whole number and a fraction. The whole number is a digit; the procedure is repeated on the new fraction. This procedure is repeated until the fractional portion is zero. If the procedure does not terminate, then the result is a repeating fraction. The first digit calculated is the most significant digit.

Example:

Converting $.1625_{10}$ to binary:

| | | |
|---|---|---|
| 0.1625•2 = 0.3250 | whole portion = 0 | MSB |
| 0.3250•2 = 0.65 | whole portion = 0 | |
| 0.65•2 = 1.3 | whole portion = 1 | |
| 0.3•2 = 0.6 | whole portion = 0 | |
| 0.6•2 = 1.2 | whole portion = 1 | |
| 0.2•2 = 0.4 | whole portion = 0 | |
| 0.4•2 = 0.8 | whole portion = 0 | |
| 0.8•2 = 1.6 | whole portion = 1 | |
| 0.6•2 = 1.2 | whole portion = 1 | |

Here we see that the fraction will repeat, since we have already multiplied 0.6 earlier. Thus

$$0.1625_{10} = 0.00101001100110011..._2$$

For mixed numbers, it is necessary to calculate the whole and fractional portions separately. Thus, for example, we know that

$$61.1625_{10} = 111101.0010100110011..._2$$

These are actually general procedures which can be used to convert a decimal number into any base, and vice versa.

Examples:

1. Converting $321.54_8$ to decimal:

$$Y = 3 \cdot 8^2 + 2 \cdot 8^1 + 1 \cdot 8^0 + 5 \cdot 8^{-1} + 4 \cdot 8^{-2}$$
$$= 192 + 16 + 1 + .625 + .0625$$
$$= 209.6875$$

$$321.54_8 = 209.6875_{10}$$

2. Converting $106.10375_{10}$ to octal:

| | | |
|---|---|---|
| 106/8 = 13 | remainder = 2 | LSB |
| 13/8 = 1 | remainder = 5 | |
| 1/8 = 0 | remainder = 1 | MSB |

Thus the whole portion is $151_8$.

| | | |
|---|---|---|
| 0.10375•8 = 0.83 | whole portion = 0 | MSB |
| 0.83•8 = 6.64 | whole portion = 6 | |
| 0.64•8 = 5.12 | whole portion = 5 | |
| 0.12•8 = 0.96 | whole portion = 0 | |
| 0.96•8 = 7.68 | whole portion = 7 | |
| 0.68•8 = 5.44 | whole portion = 5 | |

At this point we have enough significant digits. We could continue either until the procedure terminated, or until the pattern started repeating. However, those last digits are not likely to be significant. Thus we can approximate by saying that...

$$106.10375_{10} = 152.065075_8$$

3. Converting $31F.A2_{16}$ to decimal:

$$Y = 31F.A2_{16}$$
$$= 3 \cdot 16^2 + 1 \cdot 16^1 + 15 \cdot 16^0 + 10 \cdot 16^{-1} + 2 \cdot 16^{-2}$$
$$= 768 + 16 + 15 + 0.625 + 0.0078125$$
$$= 799.6328125$$

$$31F.A2_{16} = 799.6328125_{10}$$

4. Converting $7689.100854_{10}$ to hexadecimal:

| | | |
|---|---|---|
| 7689/16 = 480 | remainder = 9 | LSB |
| 480/16 = 30 | remainder = 0 | |
| 30/16 = 1 | remainder = E | |
| 1/16 = 0 | remainder = 1 | MSB |

Thus the whole portion is $1E09_{16}$.

| | | |
|---|---|---|
| 0.100854•16 = 1.613664 | whole portion = 1 | MSB |
| 0.613664•16 = 9.818624 | whole portion = 9 | |
| 0.818624•16 = 13.097984 | whole portion = D | |
| 0.097984•16 = 1.567744 | whole portion = 1 | |
| 0.567744•16 = 9.083904 | whole portion = 9 | |
| 0.083904•16 = 1.342464 | whole portion = 1 | |

Again, we likely have enough digits at this point. The exact fraction could be either very long or a long repeating pattern. For our purposes, we can approximate the overall result as:

$$7689.100854_{10} = 1E09.19D191_{16}$$

Binary <—> Octal, Hexadecimal

Converting between the binary-related systems is very easy. The procedure consists of dividing the binary digits into groups, and replacing each group with an appropriate digit. For this reason, octal and hexadecimal numbers are often used to shorten long binary numbers.

To convert from binary to octal, group the digits by three, starting on each side of the binary point, and then convert each group of three digits into its corresponding octal digit. Leading and trailing zeroes may have to be added to the left of the whole portion and the right of the fractional portion, respectively, to make complete groups of three binary digits.

Example:

Converting $11011010110101.001001101_2$ to octal:

Divide into groups of three digits:

011 011 010 110 101. 001 001 101
3 3 2 6 5 . 1 1 5

Thus $11011010110101.001001101_2 = 33265.115_8$

To convert from binary to hexadecimal, the digits are divided into groups of four digits, and then given their corresponding hexadecimal digits. Again, leading and/or trailing zeroes may be needed.

Example:

Converting $100101011101100.110110001_2$ to hexadecimal:

Divide into groups of four digits:

0100 1010 1110 1100 .1101 1000 1000
4 A E C . D 8 8

Thus $100101011101100.110110001_2 = 4AEC.D88_{16}$

To convert from octal or hexadecimal to binary, merely expand each digit into its corresponding binary representation.

Examples:

1. Convert $7324.34_8$ to binary:

```
  7   3   2   4 . 3   4
111 011 010 100. 011 100
```

Thus $7324.34_8 = 111011010100.0111_2$

2. Convert $1A2.3F5_{16}$ to binary:

```
   1    A    2  . 3    F    5
0001 1010 0010. 0011 1111 0101
```

Thus $1A2.3F5_{16} = 110100010.001111110101_2$

## Binary Arithmetic

Positive binary arithmetic is very simple, and completely analogous to decimal arithmetic. However, if we are restricted to positive numbers, then we are also restricted to addition. We need a means of representing negative numbers. Using a dash ('-') is unacceptable for representation in a computer. There are two general schemes which can be used. In binary systems, they are referred to as *1's complement* and *2's complement* representation, although they can be generalized for any base system as *diminished-radix complement* and *radix complement* representation.

## 1's Complement Representation

The one's complement of a binary number can be calculated by inverting all of the bits of the number. Fractions are handled exactly the same way, although this is convenient only for fixed-point arithmetic. Floating-point arithmetic requires other methods, which will not be discussed here.

Example:

Finding the one's complement of 110111.0101:

```
110111.0101
001000.1010     (Inverting each bit)
```

Thus the one's complement of 110111.0101 is 001000.1010.

The sign of a number is determined by the most significant bit. If the MSB is 0 the number is positive; if the MSB is 1, then the number is negative. Zero is represented by all bits being zero. However, one normally thinks of zero as being its own complement. But if we take the one's complement of zero,

```
0000
1111
```

we see that 1111 is another representation of zero. Thus, in an eight-bit representation, positive numbers range from 00000001 to 01111111; negative numbers range from 10000000 to 11111110. Note that there are just as many negative numbers as positive numbers. This eight-bit code allows us to represent the numbers from –127 to +127.

When performing addition with one's complement numbers, it is important to watch for overflow results. Whenever an overflow occurs, a correction must be made by adding 1 to the result.

In some cases, the results of an operation will not be meaningful, since the intended result cannot be represented. For instance, in the eight-bit system above, adding 127 to 127 will give a meaningless result, since 254 cannot be represented in this system. Thus the operation must be evaluated to ensure that the result is meaningful.

Examples:

All examples will use 4-bit systems. Thus the range of representable numbers is from –7 to +7.

Add 3 + 2:

```
  0011      3
+ 0010    + 2
  0101      5   result meaningful
```

Add 7 + 7 (14 cannot be represented):

```
  0111      7
+ 0111    + 7
  1110     –1   result meaningless
```

Subtract 3 from 7:

```
  0111      7
+ 1100    + –3
 10011          overflow—add 1, discard overflow bit
   + 1
  0100      4
```

Subtract 5 from 2:

```
  0010      2
+ 1010    + –5
  1100     –3   result meaningful
```

Subtract 6 from –5 (–11 cannot be represented):

```
  1010     –5
+ 1001    + –6
 10011          overflow—add 1, discard overflow bit
   + 1
  0100      4   result meaningless
```

Subtract 5.25 from 3.5 (fixed point; requires 6 bits):

```
 0011.10   3.5
+ 1010.10 + –5.25
 1110.00  –1.75   result meaningful
```

Subtract 7 from 7:

```
  0111      7
+ 1000    + –7
  1111      0   one of the representations of 0
```

The advantage of one's complement code is the fact that it is easy to compute the complement. However, the fact that there are two representations for zero is a problem. In addition, the results of subtraction frequently have to be adjusted for overflow by adding 1.

## 2's Complement Representation

The two's complement of a binary number is more difficult to calculate. It is generated by taking the one's complement, and then adding 1. Any overflow is discarded. Fractions are again handled in the same way, although 1 is added to the least significant bit.

Example:

Finding the two's complement of 110111.0101:

```
 110111.0101
 001000.1010      (take one's complement)
       + 1
 001000.1011
```

Thus the one's complement of 110111.0101 is 001000.1011.

The sign of a number is again determined by the most significant bit. If the MSB is 0 the number is positive; if the MSB is 1, then the number is negative. Zero is represented by all bits being zero. In this case, if we take the two's complement of zero, we get:

```
 0000
 1111
  + 1
 0000      (overflow is discarded)
```

giving only one representation for zero.

Thus, in an eight-bit representation, positive numbers range from 00000001 to 01111111; negative numbers range from 10000000 to 11111111. This means that there is one more negative number than there are positive numbers. So this eight-bit code allows us to represent the numbers from –128 to +127.

Addition is handled in the same fashion as with one's complement code, except that when an overflow occurs, the overflow bit is disregarded. No correction must be made to the results.

After any operation, one must still make sure that the results are meaningful.

Examples:

Add 3 + 2:

```
 0011      3
+0010     +2
 0101      5     result meaningful
```

Add 7 + 7 (14 cannot be represented):

```
 0111      7
+0111     +7
 1110     –2     result meaningless
```

Subtract 3 from 7:

```
 0111      7
+1101     +–3
10100      4     overflow—discard overflow bit
```

Subtract 5 from 2:

```
 0010      2
+1011     +–5
 1101     –3     result meaningful
```

Subtract 6 from –5 (–11 cannot be represented):

```
 1011     –5
+1010     +–6
10101      5     overflow—discard overflow bit
                 result meaningless
```

Subtract 5.25 from 3.5 (fixed point; requires 6 bits):

```
 0011.10     3.5
+1010.11    +–5.25
 1110.01    –1.75     result meaningful
```

Subtract 7 from 7:

```
 0111      7
+1001     +–7
10000      0     overflow—disregard overflow bit
```

The benefits of two's complement lie in the fact that there is only one representation for zero, and the fact that the results of operations never need adjusting due to overflow. The disadvantage is the fact that it is harder to generate the two's complement of a number.

# Signal Polarity

The polarity of signals, simple as it seems, turns out to be a potentially confusing issue. With such phrases as positive and negative logic, active HIGH and active LOW and with one person saying "asserted", another saying "active" , and another saying "enabled", all of which may or may not be well defined, it is very difficult to explain the relationships between signals. This can also make the generation of the design file more difficult.

In an attempt to sidestep the ambiguities in the language, this discussion contains tables instead of vague descriptions. The tables list the various possibilities, and explicitly state how PALASM software expects to see the equations. If you know what you want, you should be able to find how to specify your equations from the tables. The issues of input signal polarity, output signal polarity, and feedback signal polarity are treated separately.

## Input Pin Polarity

Table 1 shows the relationships between the input pin names and the use of the input in a Boolean equation. As an example of how this table can be used, if you have a signal called /A on your schematic, and you wish for the output to go HIGH when both /A and B are HIGH, then from the second row of Table 1, declare the pins as /A and B in the pin list of the design file, and use the equation

```
X = /A*B
```

The basic function A*B has been used throughout for the purpose of illustration. The same procedure holds regardless of the waveforms being used or generated.

## Output Pin Polarity

The issue of output polarity is slightly more complicated because of the issue of active-HIGH and active-LOW outputs. The possibilities are shown in Table 2. As an example, if a signal X is to go LOW only when inputs A and B are HIGH, and this function is to be implemented in an active-HIGH device, then from the third row of Table 2, declare the output pin as X in the pin list, and use the equation

```
X = /(A*B)
```

## Feedback Polarity

Using feedback combines some of the polarity issues of inputs with some of the polarity issues of outputs. It is more difficult to use a simple example for this type of circuit. In Table 3, an output is assumed to be fed back to itself. The basic principles can be extended to any output feeding back to any other output. The waveform shows the output level that is considered to be "TRUE", or "active".

As an example, if the equation for a pin /X has to contain the inverse of the output, the output signal is to be active when HIGH, and an active-LOW device is to be used, then from the sixth row of Table 3, declare the output pin as /X in the pin list, and specify the Boolean expression as:

```
/X := f(A, X)
```

meaning that the Boolean equation uses X as an input term.

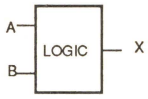
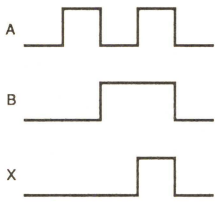
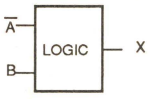
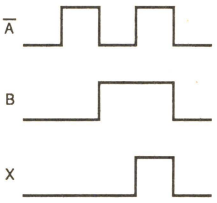
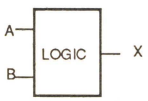
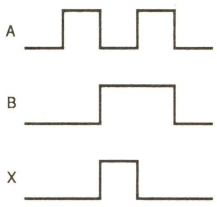
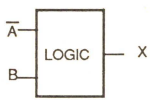
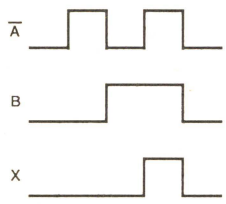
| CIRCUIT | | PALASM SOFTWARE | |
|---|---|---|---|
| Schematic | Desired Waveform | Input pin Definition in Pin List | Boolean Equation |
|  |  | A, B | X = A*B |
|  |  | /A, B | X = /A*B |
|  |  | A, B | X = /A*B |
|  |  | /A, B | X = A*B |

Table 1. Input Pin Polarity

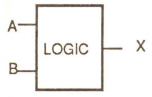
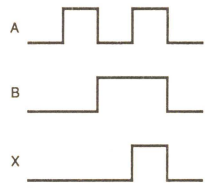
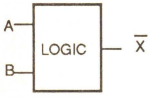
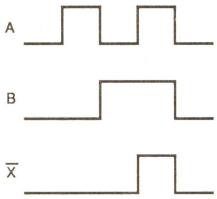
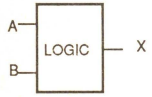
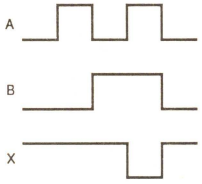
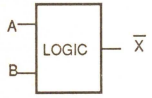
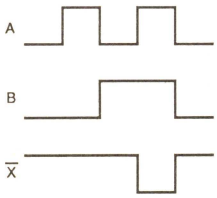
| CIRCUIT | | PALASM SOFTWARE | | DEVICE |
|---|---|---|---|---|
| Schematic | Desired Waveform | Output pin Definition in Pin List | Boolean Equation | Device Restriction |
|  |  | X<br>X | X = A*B<br>/X = /(A*B) | Active-HIGH devices<br>Active-LOW devices |
|  |  | /X<br>/X | /X = A*B<br>X = /(A*B) | Active-HIGH devices<br>Active-LOW devices |
|  |  | X<br>X | X = /(A*B)<br>/X = A*B | Active-HIGH devices<br>Active-LOW devices |
|  |  | /X<br>/X | /X = /(A*B)<br>X = A*B | Active-HIGH devices<br>Active-LOW devices |

Table 2. Output Pin Polarity

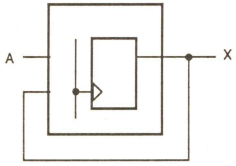
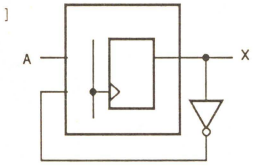
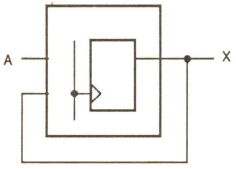
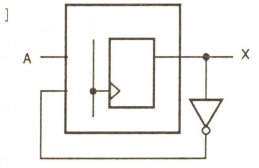
| CIRCUIT | | PALASM SOFTWARE | | DEVICE |
|---|---|---|---|---|
| Schematic | Desired Waveform | Output pin Definition in Pin List | Boolean Equation | Device Restriction |
|  |  | X<br>X | X = f(A,X)<br>/X = /f(A,X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | X<br>X | X = f(A,/X)<br>/X = /f(A,/X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | X<br>X | X = /f(A,X)<br>/X = f(A,X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | X<br>X | X = /f(A,/X)<br>/X = f(A,/X) | Active-HIGH devices<br>Active-LOW devices |

Table 3. Feedback Signal Polarity

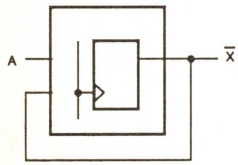
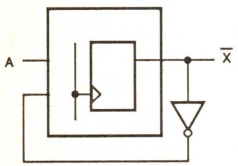
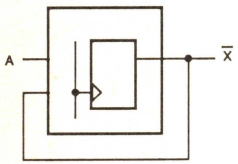
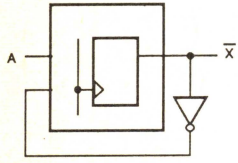
| CIRCUIT | | PALASM SOFTWARE | | DEVICE |
|---|---|---|---|---|
| Schematic | Desired Waveform | Output pin Definition in Pin List | Boolean Equation | Device Restriction |
|  |  | /X<br>/X | /X = f(A,/X)<br>X = /f(A,/X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | /X<br>/X | /X = f(A,X)<br>X = /f(A,X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | /X<br>/X | /X = /f(A,/X)<br>X = f(A,/X) | Active-HIGH devices<br>Active-LOW devices |
|  |  | /X<br>/X | /X = /f(A,X)<br>X = f(A,X) | Active-HIGH devices<br>Active-LOW devices |

**Table 3. Feedback Signal Polarity** (Cont'd.)

6

# Glossary

**10KH** (adj.) A family of ECL devices. Circuits are temperature compensated. See also: ECL, 100K, temperature compensation.

**100K** (adj.) A family of ECL devices. Circuits are both temperature and voltage compensated. They have lower power dissipation and higher speed than their 10KH counterparts. See also: ECL, temperature compensation, voltage compensation, power dissipation, 10KH.

## A

**active high** (adj.) See polarity.

**active low** (adj.) See polarity.

**ALS** (adj.) Advanced Low-power Schottky TTL family. Characterized as a lower power version of the AS family, and actually faster and lower power than the LS family. See also: AS, LS, TTL, Schottky TTL.

**AND** 1. (adj.) One of the three elementary logic functions. Result of the AND operation is true if and only if all operands are true. 2. (v.t.) To perform the AND operation.

**AS** (adj.) Advanced Schottky TTL family. High-speed versions of the standard Schottky TTL family. Generally use oxide isolated technology for very high speed. See also: Schottky TTL, TTL, oxide isolation.

**assertive high** (adj.) Same as "active high". See polarity.

**assertive low** (adj.) Same as "active low". See polarity.

**astable** (adj.) Describes a system which has no stable state. Such a system will oscillate. Astable circuits can be used to generate timing and synchronizing clock signals. See also: bistable, monostable.

**asynchronous** 1. (adj.) Describes a sequential logic system wherein operations are not synchronized to a common clock. 2. (adj.) Describes signals whose behavior and timing are completely unrelated to a particular clock. Such signals can either be random or based on another clock which has a different frequency. 3. (adj.) Describes a communication protocol whereby the timing of various operations is not determined by a system clock, but rather by events whose relationships are known, but whose exact timing cannot be precisely predicted. See also: sequential, clock, synchronous.

## B

**binary** (adj.) Having only two possible states, which can be variously called on/off, 1/0, true/false, high/low, etc.

**bipolar** (adj.) One of the two basic types of transistor. In logic design, used for TTL, ECL, and $I^2L$ families. See also: TTL, ECL, $I^2L$, MOS.

**bistable** (adj.) Describes a system which has 2 stable states. Any other state is unstable, and will eventually change to one of the stable states. A flip-flop is the most common electronic bistable circuit. See also: flip-flop, astable, monostable.

**bit** 1. (n.) Binary Digit. One unit of binary information. 2. (n.) A measure of the storage capacity of a memory chip. See also: binary.

**blank** (adj.) Describes the state of a programmable cell after manufacturing, and before any programming, or, in the case of an erasable device, after erasure. Opposite of "programmed". See also: programmable cell, programmed, program, erase.

**buffer** (n.) A logic gate which performs the logic identity function; i.e., the input is passed through unchanged. Used to isolate various parts of a system, or to provide voltage or current amplification.

## C

**chip** (n.) A single piece of semiconductor material which contains an integrated circuit. Sometimes called a die if not in a package. See also: integrated circuit, die, package.

**clock** 1. (adj.) A signal used to synchronize the operation of a system. 2. (adj.) An input to a clocked flip-flop. The flip-flop will not change state until an appropriate pulse appears at the clock input. 3. (n.) A circuit which generates a clock signal. 4. (v.t.) To pulse the clock signal or the clock input of a clocked flip-flop. See also: flip-flop, clocked flip-flop.

**clocked flip-flop** (n.) A flip-flop that does not change state until a clock signal is received. See also: flip-flop, unclocked flip-flop, clock.

**CMOS** (n., adj.) Complementary MOS. A type of circuit which makes use of both N-channel and P-channel MOS transistors. Many CMOS logic circuits consume no power when not actually switching. See also: MOS, NMOS, PMOS, standby power.

**combinational** (adj.) See combinatorial.

**combinatorial** (adj.) Refers to a logic circuit which implements logic functions of present input signals only. Also called combinational. See also: sequential.

**complement** 1. (adj). Refers to a signal which is identical to some reference signal, except that it is of opposite polarity. Opposite of "true". 2. (v.t.) To invert. See also: true, polarity, invert.

**complementary** (adj.) Refers to logic device outputs which implement identical logic functions, but with opposite polarities. Used on some PLDs and ECL devices. See also: polarity, PLD, ECL.

## D

**die** (n.; plural: dice) Same as a chip, particularly before being placed in a package. See also: chip, package.

**DIP** (n.) Dual In-line Package. The most common integrated circuit package. It is rectangular in shape, with widths ranging from .300 inch to .900 inch, and has vertical leads along the length. Available for up to 64 pins. See also: integrated circuit, package.

**disable** 1. (v.t.) To turn off a three-state output. 2. (v.t.) To inhibit another function, such as "disabling the clock". See also: three-state, enable.

**download** 1. (v.t.) To pass data from one machine to a less complex machine. 2. (n.) The act of downloading data. See also: upload.

## E

**ECL** (n., adj.) Emitter Coupled Logic family. An extremely high speed family of bipolar logic and memory devices. See also: bipolar.

**EE cell ($E^2$ cell)** (n.) A floating gate cell which can be both programmed and erased with electrical signals.

**EEPROM** (n.) Electrically Erasable Programmable Read-Only Memory. A nonvolatile read-only memory device which can be erased and reprogrammed, both with special electrical signals. See also: program, erase, EPROM, PROM, ROM, RAM, non-volatile.

**enable** 1. (v.t.) To turn on a three-state output. 2. (adj.) By itself, usually refers to a pin which is used to enable a three-state output. Also called "output enable". 3. (adj.) Used with other function names, indicates a qualifier or inhibitor of the function. For example, "clock enable" is a function which qualifies the clock function. 4. (v.t.) To allow a signal which has been disabled to function; for example, "enabling the clock" removes any restraint which may disable the clock signal. See also: three-state, disable.

**EPROM** (n.) Erasable Programmable Read-Only Memory. A non-volatile read-only memory device which can be erased and reprogrammed. Erasure is accomplished by exposing the die to ultraviolet light for a period of time. Die must be packaged in a windowed package to allow erasure. See also: program, erase, EEPROM, PROM, ROM, RAM, non-volatile, windowed package.

**erase** 1. (v.t.) To return a programmed device to its blank state. Opposite of "program". 2. (v.t.) To return an individual programmable cell to its blank state. See also: blank, programmable cell, program.

## F

**finite state machine (FSM)** (n.) A machine which can be in one of a finite number of states. Often used for logic circuits which sequence through various states. Such a circuit is referred to as sequential. See also: sequential.

**flip-flop** (n.) A bistable digital circuit. The simplest variety is called an S-R flip-flop. Other types are J-K, T, and D-type. May be unclocked or clocked. See also: bistable, unclocked flip-flop, clocked flip-flop.

**floating gate** (n.) A gate on an MOS transistor which is not connected to anything. Used to store charge; forms the basis of UV cells and EE cells. See also: MOS, gate, UV cell, EE cell.

**FPGA** (n.) Field Programmable Gate Array. An array of logic gates whose configuration can be programmed by the customer. The gates are often NAND gates, but can also be NOR gates. See also: gate, program, NAND, NOR.

**FPLA** (n.) Field Programmable Logic Array. See PLA.

**FPLS** (n.) Field Programmable Logic Sequencer. A programmable logic device which is intended for sequencing or state machine applications. See also: finite state machine.

**functionally complete** (adj.) Refers to a logic operation or group of operations from which any complex logic function can be built. The NAND and NOR operators are functionally complete. See also: NAND, NOR.

**fuse** (n.) As used in programmable logic, usually refers to a lateral metal link fuse. See also: lateral fuse.

**fuse map** (n.) A graphic representation of the contents of a PLD. The state of each connection (fuse or other programmable cell) is represented, usually with "X" indicating an intact connection, and "-" indicating an open connection. See also: PLD, programmable cell.

## G

**gate** 1. (n.) A fundamental logic element. The elementary gates provide NOT, AND, and OR logic functions. 2. (n.) The control terminal of a gated D-type latch. See also: latch, gated latch.

**gate array** (n.) A logic device which consists of an array of logic gates (usually NAND) which can be interconnected during fabrication. A custom metallization pattern is used to configure the desired functions. See also: gate, NAND, metallization.

**gate equivalency** (n.) A rough measure of the complexity of a digital logic integrated circuit. Indicates the approximate number of discrete logic gates that would be needed to implement the same function. See also: gate.

**gated latch** (n.) Generally refers to an unclocked D-type flip-flop which has a control signal called a gate. When the gate is "open", the flip-flop output follows the data input. When the gate is "closed", the output holds its current state. Also called a transparent latch. See also: flip-flop, unclocked flip-flop, gate, latch.

## H

**HAL® device** (n.) Hard Array Logic device. A version of a PAL device which is configured during fabrication with a custom metallization pattern. HAL is a registered trademark of Monolithic Memories. See also: PAL device, metallization.

## I

**$I^2L$ (IIL)** (n., adj.) Integrated Injection Logic. A less common bipolar logic design technique which, when used, is found primarily in portions of LSI and VLSI circuits. See also: bipolar, LSI, VLSI.

**integrated circuit** (n.) An electronic device which has many transistors and other semiconductor components integrated onto one piece of silicon. Often abbreviated IC.

**invert** (v.t.) To perform the logical NOT function on a digital signal. To reverse the polarity of a digital signal. See also: polarity, NOT.

**inverter** (n.) A logic gate which performs logical inversion, or the NOT operation. See also: gate, NOT.

**I/O (Input/Output)** 1. (n.) The methods and equipment used to pass information into and/or out of a system or device. 2. (adj.) On a programmable logic device, a pin which can function as an input and/or an output.

## J

**JEDEC** 1. (n.) Joint Electronic Device Engineering Council. A council which creates, approves, arbitrates, and/or oversees industry standards for electronic devices. 2. (adj.) In programmable logic, refers to a computer file containing information about the programming of a device. The file format is a JEDEC-approved standard. Used for downloading to programmers. See also: program, programmer, download.

**junction isolation** (n.) A bipolar integrated circuit fabrication technique which uses P-N functions to isolate transistors. This is the original integrated circuit technology, and is being supplanted by oxide isolation in places where speed is critical. See also: oxide isolation, bipolar.

## K

**Karnaugh map (K-map)** (n.) A graphic tool for minimizing sum-of-products or product-of-sums logic functions. Useful for up to six logic variables. See also: sum-of-products, product-of-sums.

## L

**latch** 1. (n.) A type of flip-flop. Means different things to different people. In general, an unclocked flip-flop. Sometimes used to refer specifically to a gated D-type flip-flop. 2. (v.t.) To capture a signal in a latch. See also: flip-flop, unclocked flip-flop, gate, gated latch.

**latch up** (v.t.) To enter the latch-up condition. See also: latch-up.

**latch-up** (n.) A condition in which a circuit draws uncontrolled amounts of current, and certain voltages are forced, or "latched-up" to some level. Used especially in reference to CMOS devices, which can latch up if the operating conditions are violated. See also: CMOS, latch up.

**lateral fuse** (n.) A thin metal link which is disconnected when programmed. Connected in the blank state, disconnected in the programmed state. Usually just called a "fuse". See also: program, programmed, blank.

**LCC** (n.) Leadless Chip Carrier. A ceramic integrated circuit package having no leads. Connection is made to metal contacts which are flush with the package. See also: integrated circuit, lead, package.

**lead** (n.) [lēd] A metal conductor which provides a connection from the inside of an integrated circuit package to the outside world for soldering or other mounting techniques. See also: integrated circuit.

**logic array** (n.) Generally an array of programmable cells which attach inputs to logic gates of a specified type. See also: program, gate, programmable cell.

**logic simulation** (n.) A means whereby a logic design can be evaluated on a computer before actually being built. The computer simulates the behavior of the components to predict the behavior of the overall circuit.

**LS** (adj.) Low-power Schottky TTL family. Lower power version of the standard Schottky TTL family. See also: TTL, Schottky TTL.

**LSI** (adj.) Large-Scale Integration. A rough measure of the complexity of a digital circuit. Characterized as having 100–5000 gate equivalents for logic chips, or 1K–16K bits for memory chips. See also: gate equivalent, bit, VLSI, SSI, MSI.

## M

**maxterm** (n.) A sum in the canonical product-of-sums form. Each maxterm contains every input variable, in either true or complemented form. See also: product-of-sums, true, complement.

**metallization** (n.) The process of connecting the various elements of an integrated circuit or printed circuit board by placing a layer of metal over the entire wafer or board, and then selectively etching away unwanted metal. A photolithographic mask defines the pattern of connections. See also: integrated circuit, wafer, printed circuit board.

**minterm** (n.) A product in the canonical sum-of-products form. Each minterm contains every input variable, either in true or complemented form. See also: sum-of-products, true, complement.

**monolithic** (adj.) In the electronics industry, refers to a circuit which has been integrated onto one semiconductor chip. Integrated circuits are monolithic by definition. See also: integrated circuit.

**monostable** (adj.) Describes a system which has 1 stable state. Any other state is unstable, and will eventually change to the stable state. The most common monostable circuit is a "one-shot". See also: bistable, astable.

**MOS** (n., adj.) Metal-Oxide-Semiconductor transistor. One of the two basic types of transistor. In logic design, used for NMOS, PMOS, and CMOS families. See also: NMOS, PMOS, CMOS, bipolar.

**MSI** (adj.) Medium-Scale Integration. A rough measure of the complexity of a digital logic circuit. Characterized as having 10–100 gate equivalents. See also: gate equivalent, SSI, LSI, VLSI.

## N

**NAND** (adj.) Not AND. A commonly used logic gate which is equivalent to an AND gate followed by an inverter. The NAND logic operation is functionally complete. See also: gate, inverter, functionally complete, AND.

**negative logic** (n.) A physical implementation of logic wherein a low voltage level represents a logic 1, or "true", and a high voltage level represents a logic 0, or "false". See also: positive logic, polarity.

**NMOS** (n., adj.) N-channel MOS. A type of circuit which makes exclusive use of N-channel MOS transistors. See also: MOS, PMOS, CMOS.

**non-volatile** (adj.) Refers to memory devices which do not lose their contents when power is removed. See also: volatile.

**NOR** (adj.) Not OR. A logic gate which is equivalent to an OR gate followed by an inverter. The NOR logic operation is functionally complete. See also: gate, inverter, functionally complete, OR.

**NOT** (adj.) One of the three elementary logic functions. Unary operation whose result is true if and only if the operand is false.

## O

**OR** 1. (adj.) One of the three elementary logic functions. Result of the OR operation is false if and only if all operands are false. 2. (v.t.) To perform the OR operation.

**OTP** (adj.) One-Time Programmable. Refers to programmable devices which are UV-erasable, but which are not packaged in windowed packages. As a result, there is no way to erase the device, making it programmable only once. See also: program, erase, UV-erasable, windowed package.

**oxide isolation** (n.) A bipolar integrated circuit fabrication technique which uses silicon dioxide to isolate transistors. This results in higher speed and density. See also: junction isolation, bipolar.

## P

**package** (n.) The encasement which protects a die and provides convenient electrical contact to the die. Materials used are generally ceramic or plastic compounds. There is a variety of shapes and sizes. See also: die.

**PAL® device** (n.) Programmable Array Logic device. A PLD which implements logic via a programmable AND logic array driving a fixed OR logic array. PAL is a registered trademark of Monolithic Memories. See also: program, logic array, sum-of-products, PLD, AND, OR.

**PLA** (n.) Programmable Logic Array. A programmable logic device which implements sum-of-products logic via a programmable AND logic array driving a programmable OR logic array. See also: program, logic array, sum-of-products, AND, OR.

**PLCC** (n.) Plastic Leaded Chip Carrier. A molded plastic integrated circuit package with leads shaped like a "J" (J-leads). Intended for surface mounting. See also: integrated circuit, lead, surface mounting, package.

**PLD** (n.) Programmable Logic Device. Generic term for a logic device whose function can be configured by the customer after purchase. See also: program.

**PMOS** (n., adj.) P-channel MOS. A type of circuit which makes exclusive use of P-channel MOS transistors. See also: MOS, NMOS, CMOS.

**polarity** (n.) Specifies the sense of "active" and "inactive", or "true" and "false" in a digital signal. "Active high" represents "true" as a high signal; "active low" represents "true" as a low signal.

**positive logic** (n.) A physical implementation of logic wherein a high voltage level represents a logic 1, or "true", and a low voltage level represents a logic 0, or "false". See also: negative logic, polarity.

**power dissipation** (n.) The amount of electrical power used by a device. Calculated as the product of the operating voltage and current. Measured in watts (W) or milliwatts (mW), as appropriate. Sometimes incorrectly used to refer to the operating current only.

**printed circuit board (PC board, PCB)** (n.) A board for assembling electrical components. Component connections are made by metal traces which have been fabricated through a metallization process. See also: trace, metallization.

**product-of-sums (POS)** (adj.) A representation of a logic function where the input signals are individually inverted (if necessary), then ORed together to form sums which are ANDed together. Any combinatorial logic function can be represented in product-of-sums form. See also: sum-of-products, combinatorial, AND, OR.

**product term (pterm, p-term)** (n.) An AND gate in a PLD which implements sum-of-products logic. See also: sum-of-products, PLD, AND, gate.

**product term sharing** (n.) See product term steering.

**product term steering** (n.) A means whereby product terms in a PAL device can be routed to one of two device outputs, instead of being dedicated only to one output. Sometimes called "product term sharing". See also: product term, PAL device.

**program** 1. (v.t.) As used in programmable logic, to configure a blank device so that it can perform some desired function. Applies to memory and logic devices. Opposite of "erase". 2. (v.t.) To change an individual programmable cell from a blank state to a programmed state. See also: blank, programmable cell, programmed, erase.

**programmable cell** (n.) Any of a variety of cells which can be altered by applying certain electrical signals. Various types are lateral and vertical fuses, UV cells, $E^2$ cells, and even RAM cells. All but RAM cells are non-volatile. See also: lateral fuse, vertical fuse, UV cell, $E^2$ cell, RAM cell, non-volatile, volatile.

**programmed** (adj.) Describes the state of a programmable cell or device after programming. Opposite of "blank".

**programmer** (n.) A device or machine used for configuring, or "programming", PLDs or PROMs. See also: program, PLD, PROM.

**PROM** (n.) Programmable Read-Only Memory. A nonvolatile memory device whose contents are programmed by the customer. Once programmed, it cannot be erased. Also functions as a PLD with a fixed AND logic array which drives a programmable OR logic array. See also: program, erase, EEPROM, EPROM, ROM, RAM, non-volatile, AND, OR, logic array.

## R

**RAM** (n.) Random-Access Memory. Sometimes called read/write memory. A type of memory device which can be written to and read at any time. Such memory is volatile. Actually a misnomer, since most types of memories can be accessed randomly. The distinguishing feature is the fact that RAM is designed specifically to be written to in normal usage. See also: ROM, volatile.

**RAM cell** (n.) A cell which is used make one bit of volatile memory in a RAM. Can also form the basis of a programmable logic connectivity array. See also: RAM, volatile.

**ROM** (n.) Read-Only Memory. A nonvolatile memory device which has its contents defined when manufactured. No changes can be made to the memory contents. See also: PROM, EPROM, EEPROM, RAM, non-volatile.

## S

**Schottky TTL** (adj.) Family of TTL devices which make use of Schottky diodes for higher speed. See also: TTL.

**security fuse** (n.) A PLD feature which allows a user to "secure" the PLD after programming. This prevents subsequent copying of the contents of the PLD. See also: PLD, program.

**semicustom** (adj.) Refers to a circuit which has been partially designed by the device vendor, and partially designed, or configured, by the customer. Primary types are PLDs, gate arrays, and standard cell circuits. See also: PLD, gate array, standard cell.

**sequential** (adj.) Refers to a logic circuit whose operation depends both on present input signals and previous operations, or states. Requires some kind of memory (usually flip-flops) for remembering past states. See also: flip-flop, combinatorial.

**SSI** (adj.) Small Scale Integration. A rough measure of the complexity of a digital logic circuit. Characterized as having less than 10 gate equivalents. See also: gate equivalent, MSI, LSI, VLSI.

**standard cell** (n.) A method of designing semicustom or full custom circuits whereby predefined cells are brought together to provide the specified function. Unlike gate arrays, all fabrication steps are customized, instead of just the metallization step. See also: semicustom, gate array, metallization.

**standby power** (n.) The power consumed by a device when none of the device inputs are switching. Usually used in reference to CMOS devices, many of which consume practically no standby power. See also: CMOS.

**sum-of-products (SOP)** (adj.) A representation of a logic function where the input signals are individually inverted (if necessary), then ANDed together to form products which are ORed together. Any combinatorial logic function can be represented in sum-of-products form. See also: product-of-sums, combinatorial, AND, OR.

**surface mounting** (n.) A printed circuit board assembly technique whereby the integrated circuit packages are placed on the board with no leads protruding through to the other side. Packages can thus be mounted on both sides of the board. See also: printed circuit board, lead, through-hole mounting.

**synchronous** 1. (adj.) Describes a sequential logic system wherein all operations are synchronized to a common clock. 2. (adj.) Describes signals whose behavior and timing are synchronized to a clock. 3. (adj.) Describes a communication protocol whereby the timing of various operations is determined by a system clock. See also: sequential, clock, asynchronous.

## T

**temperature compensation** (n.) A circuit feature which allows some electrical characteristics to remain relatively constant with some variation in operating temperature.

**three-state** (adj.) A type of logic device output which can be in one of three-states: HIGH, LOW, and OFF, or High-Z (high impedance). When enabled (on), performs as a normal binary output. When disabled (off), acts as an open pin. See also: enable, disable, binary.

**through-hole mounting** (n.) A printed circuit board assembly technique whereby the leads of the various components extend through holes in the board. These leads are then soldered from the opposite side of the board. See also: printed circuit board, lead, surface mounting.

**trace** 1. (n.) During logic simulation, the behavior of a signal or group of signals. The results can sometimes be stored in a "trace file" on disk for later analysis. 2. (n.) A thin layer of metal on a printed circuit board which provides connections between components. Performs the function of a wire. See also: logic simulation, printed circuit board.

**transparent latch** (n.) See gated latch.

**TRI-STATE®** (adj.) See three-state. TRI-STATE is a registered trademark of National Semiconductor Corp.

**true** (adj.) Refers to a signal which is identical to some reference signal, with the same polarity. Opposite of "complement". See also: complement, polarity.

**TTL** (adj.) Transistor-Transistor Logic family. The most widely used family of bipolar logic devices. The name refers to the particular circuit design technique used. See also: bipolar.

## U

**unclocked flip-flop** (n.) A flip-flop that changes state as soon as the appropriate control signals are applied. See also: flip-flop, clocked flip-flop.

**upload** 1. (v.t.) To pass data from one machine to a more complex machine. 2. (n.) The act of uploading data. See also: download.

**UV cell** (n.) A floating gate cell which can be erased by exposure to ultraviolet (UV) light. See also: floating gate, erase.

**UV-erasable** (adj.) Refers to devices or programmable cells which can be erased when exposed to ultraviolet (UV) light for a period of time. See also: programmable cell, erase.

## V

**vertical fuse** (n.) A transistor arranged such that the emitter and base are shorted together when programmed. Disconnected in the blank state, connected in the programmed state. See also: program, programmed, blank.

**VLSI** (adj.) Very Large Scale Integration. A rough measure of the complexity of a digital circuit. Characterized as having 5000 or more gate equivalents for logic chips, or 16K or more bits for memory chips. See also: gate equivalent, bit, SSI, MSI, LSI.

**volatile** (adj.) Refers to memory devices which lose their contents when power is removed. See also: non-volatile.

**voltage compensation** (n.) A circuit feature which allows some electrical characteristics to remain relatively constant with some variation in the supply voltage.

## W

**wafer** (n.) A round slice of very pure silicon which is used in the fabrication of integrated circuits. Several circuits can be built on one wafer. See also: integrated circuit.

**windowed package** (n.) A package which has a quartz window in the lid directly over the die. This makes it possible to expose the die to ultraviolet light for erasing the device. See also: erase, die, package.

# Competitive Cross-Reference

## Altera

| Altera | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| EP310 | 50/40 | PALC16L8Q-25 | 25/45 | |
| | | PALC16R8Q-25 | 25/45 | |
| | | PALC16R6Q-25 | 25/45 | |
| | | PALC16R4Q-25 | 25/45 | |
| EP310-2 | 35/40 | PALC16L8Q-25 | 25/45 | |
| | | PALC16R8Q-25 | 25/45 | |
| | | PALC16R6Q-25 | 25/45 | |
| | | PALC16R4Q-25 | 25/45 | |
| | | PAL16RA8 | 30/170 | Offers asynchronous clocking |
| EP320 | 45/10* | ZHAL20A** | | |
| | | PALC16L8Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R8Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R6Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R4Z-35 | 35/10 | Available 1Q88<br>Standby supply current = 100 µA for all device types. |
| EP320-2 | 35/10* | ZHAL20A** | 25/5 | |
| | | PALC16L8Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R8Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R6Z-35 | 35/10 | Available 1Q88 |
| | | PALC16R4Z-35 | 35/10 | Available 1Q88<br>Standby supply current = 100 µA for all device types. |
| EP320-1 | 25/10* | ZHAL20A** | 25/5 | |
| | | PALC16L8Z-25 | 25/10 | Available 1Q88 |
| | | PALC16R8Z-25 | 25/10 | Available 1Q88 |
| | | PALC16R6Z-25 | 25/10 | Available 1Q88 |
| | | PALC16R4Z-25 | 25/10 | Available 1Q88<br>Standby supply current = 100 µA for all device types. |
| EP600 | 55/10* | AmPALC29M16-45 | 45/120 | |
| | | AmPALC29MA16-45 | 45/120 | |
| EP600-3 | 45/10* | PALC20L8Z-45 | 45/10 | |
| | | PALC20R8Z-45 | 45/10 | |
| | | PALC20R6Z-45 | 45/10 | |
| | | PALC20R4Z-45 | 45/10 | |
| EP600-2 | 35/10* | AmPALC29M16-35 | 35/120 | |
| | | AmPALC29MA16-35 | 35/120 | |
| | | PALC20L8Z-35 | 35/10 | |
| | | PALC20R8Z-35 | 35/10 | |
| | | PALC20R6Z-35 | 35/10 | |
| | | PALC20R4Z-35 | 35/10 | |
| | | PAL20RA10 | 30/200 | Offers asynchronous clocking. |
| | | PAL20RA10-20 | 20/200 | Available 4Q87 |
| | | PAL32VX10 | 35/180 | Offers programmable flip-flops. |
| | | PAL32VX10A | 25/180 | |

## Altera (Cont'd.)

| Altera | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| EP900<br>EP900-3<br>EP900-2<br>EP900-1 | 60/15<br>50/15<br>45/15<br>35/15 | LCA M2064 | 70 MHz/5 | *Note: EP900 has only 8 PT's per output.* |
| EP1210<br>EP1210-2<br>EP1210-1 | 90/65<br>65/65<br>50/65 | LCA M2064 | 70 MHz/5 | *Note: EP1200 does not have zero stand-by supply current when in turbo mode.* |
| EP1800-3<br>EP1800-2<br>EP1800-1 | 18.5 MHz/30<br>20.8 MHz/30<br>25 MHz/30 | LCA M2018 | 50 MHz/5 | *Note: EP1800 does not have zero stand-by supply current when in turbo mode* |

*Note: All power values given are for f = 1 MHz with no output load unless otherwise noted.*

**ZHAL20A series includes:

| | | | | |
|---|---|---|---|---|
| 16L8A | 16P8A | 10H8A | 10L8A | 16C1A |
| 16R8A | 16RP8A | 12H6A | 12L6A | |
| 16R6A | 16RP6A | 14H4A | 14L4A | |
| 16R4A | 16RP4A | 16H2A | 16L2A | |

## Cypress

| Cypress | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| PALC16L8L-25<br>PALC16R8L-25<br>PALC16R6L-25<br>PALC16R4L-25 | 25/45<br>25/45<br>25/45<br>25/45 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | MMI has reduced ground bounce sensitivity by scaling down output drive to 8 mA. |
| PALC16L8-25<br>PALC16R8-25<br>PALC16R6-25<br>PALC16R4-25 | 25/70<br>25/70<br>25/70<br>25/70 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | |
| PALC16L8L-35<br>PALC16R8L-35<br>PALC16R6L-35<br>PALC16R4L-35 | 35/45<br>35/45<br>35/45<br>35/45 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | |
| PALC16L8-35<br>PALC16R8-35<br>PALC16R6-35<br>PALC16R4-35 | 35/70<br>35/70<br>35/70<br>35/70 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | |
| PLDC20G10-25<br>PLDC20G10-35 | 25/55<br>35/55 | PAL22RX8A | 25/210 | Superset of Standard and Combinatorial 24 pin devices, except RS family. Includes programmable flip-flops. |
| PALC22V10-25<br>PALC22V10-35 | 25/90<br>35/90 | PALC22V10H-25<br>PALC22V10H-35 | 25/90<br>35/90 | |

## Fairchild

| Fairchild | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed Power | |
| PAL16L8A | 25/180 | Am/PAL16L8A | 25/180 | AmPAL16L8A at 155 mA |
| PAL16R8A | 25/180 | Am/PAL16R8A | 25/180 | |
| PAL16R6A | 25/180 | Am/PAL16R6A | 25/180 | |
| PAL16R4A | 25/180 | Am/PAL16R4A | 25/180 | |
| PAL16P8A | 25/180 | AmPAL18P8A | 25/180 | |
| | | PAL16P8A | 25/30*/180 | *Polarity fuse programmed high |
| PAL16RP8A | 25/180 | PAL16RP8A | 25/30*/180 | *Polarity fuse programmed high |
| PAL16RP6A | 25/180 | PAL16RP6A | 25/30*/180 | *Polarity fuse programmed high |
| PAL16RP4A | 25/180 | PAL16RP4A | 25/30*/180 | *Polarity fuse programmed high |
| PAL16L8B | 15/180 | Am/PAL16L8B | 15/180 | |
| PAL16R8B | 15/180 | Am/PAL16R8B | 15/180 | |
| PAL16R6B | 15/180 | Am/PAL16R6B | 15/180 | |
| PAL16R4B | 15/180 | Am/PAL16R4B | 15/180 | |
| PAL16P8B | 15/180 | AmPAL18P8B | 15/180 | |
| PAL16RP8B | 15/180 | Am/PAL16R8B | 15/180 | Fairchild part has programmable polarity |
| PAL16RP6B | 15/180 | Am/PAL16R6B | 15/180 | Fairchild part has programmable polarity |
| PAL16RP4B | 15/180 | Am/PAL16R4B | 15/180 | Fairchild part has programmable polarity |
| | | AmPAL23S8-20 | 20/200 | |
| PAL16L8D | 10/180 | Am/PAL16L8D | 10/180 | |
| PAL16R8D | 10/180 | Am/PAL16R8D | 10/180 | |
| PAL16R6D | 10/180 | Am/PAL16R6D | 10/180 | |
| PAL16R4D | 10/180 | Am/PAL16R4D | 10/180 | |

## Intel

| Intel | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| 5C031 | 50/40*<br>40/40* | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | |
| 5C032 | 35/5*<br>30/5* | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25<br>PALC16L8Z-25<br>PALC16R8Z-25<br>PALC16R6Z-25<br>PALC16R4Z-25<br>PALC16L8Z-35<br>PALC16R8Z-35<br>PALC16R6Z-35<br>PALC16R4Z-35 | 25/45<br>25/45<br>25/45<br>25/45<br>25/10*<br>25/10*<br>25/10*<br>25/10*<br>35/10*<br>35/10*<br>35/10*<br>35/10* | <br><br><br><br>Available 1Q88<br>Available 1Q88<br>Available 1Q88<br>Available 1Q88<br>Available 1Q88<br>Available 1Q88<br>Available 1Q88<br>Available 1Q88 |
| 5C060 | 55/10*<br>45/10* | AmPALC29MA16-35<br>PALC20L8Z-35<br>PALC20R8Z-35<br>PALC20R6Z-35<br>PALC20R4Z-35<br>PALC20L8Z-45<br>PALC20R8Z-45<br>PALC20R6Z-45<br>PALC20R4Z-45 | 35/120<br>35/10*<br>35/10*<br>35/10*<br>35/10*<br>45/10*<br>45/10*<br>45/10*<br>45/10* | |

*f = 1 MHz

## Lattice

| Lattice | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| GAL16V8-25 | 25/45 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | GAL16V8 is a universal 20 pin PAL device architecture. |
| GAL16V8-25 | 25/90 | Am/PAL16L8B-2/AL<br>Am/PAL16R8B-2/AL<br>Am/PAL16R6B-2/AL<br>Am/PAL16R4B-2/AL | 25/90<br>25/90<br>25/90<br>25/90 | |
| GAL16V8-35 | 35/45 | PALC16L8Q-25<br>PALC16R8Q-25<br>PALC16R6Q-25<br>PALC16R4Q-25 | 25/45<br>25/45<br>25/45<br>25/45 | |
| GAL16V8-35 | 35/90 | Am/PAL16L8A-2/L<br>Am/PAL16R8A-2/L<br>Am/PAL16R6A-2/L<br>Am/PAL16R4A-2/L | 35/90<br>35/90<br>35/90<br>35/90 | AmPAL16L8L at 80 mA |
| GAL16V8-15 | 15/90 | PAL16L8H-15<br>PAL16R8H-15<br>PAL16R6H-15<br>PAL16R4H-15 | 15/90<br>15/90<br>15/90<br>15/90 | MMI—Available 4Q87<br>MMI—Available 4Q87<br>MMI—Available 4Q87<br>MMI—Available 4Q87 |
| GAL20V8-35 | 35/45 | PALC20L8Z-35<br>PALC20R8Z-35<br>PALC20R6Z-35<br>PALC20R4Z-35 | 35/10*<br>35/10*<br>35/10*<br>35/10* | GAL20V8 is a universal 24 pin PAL device. MMI ZPAL devices offer advantage of 100 µA standby current<br>*f = 1 MHz |
| GAL20V8-35 | 35/90 | PAL20L8A-2<br>PAL20R8A-2<br>PAL20R6A-2<br>PAL20R4A-2 | 35/105<br>35/105<br>35/105<br>35/105 | |
| GAL20V8-25 | 25/45 | PAL22RX8A<br>PALC22V10H-25 | 25/210<br>25/90 | |
| GAL20V8-25 | 25/90 | PAL20L8B-2<br>PAL20R8B-2<br>PAL20R6B-2<br>PAL20R4B-2<br>PALC22V10H-25 | 25/105<br>25/105<br>25/105<br>25/105<br>25/90 | |
| GAL39V18 | 30/120 | PAL32VX10A<br>PAL22RX8A<br>AmPALC29M16-35 | 25/180<br>25/210<br>35/120 | Advanced architecture PAL devices |

## National Semiconductor Corporation

| NSC | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| PAL10H8 | 35/90 | PAL10H8 | 35/90 | |
| PAL12H6 | 35/90 | PAL12H6 | 35/90 | |
| PAL14H4 | 35/90 | PAL14H4 | 35/90 | |
| PAL16H2 | 35/90 | PAL16H2 | 35/90 | |
| PAL16C1 | 40/90 | PAL16C1 | 40/90 | |
| PAL10L8 | 35/90 | PAL10L8 | 35/90 | |
| PAL12L6 | 35/90 | PAL12L6 | 35/90 | |
| PAL14L4 | 35/90 | PAL14L4 | 35/90 | |
| PAL16L2 | 35/90 | PAL16L2 | 35/90 | |
| | | AmPAL18P8L | 35/90 | |
| PAL10H8A | 25/90 | ZHAL20A | 25/5* | No direct equivalent. Can be replaced by ZHAL20A series. |
| PAL12H6A | 25/90 | AmPAL18P8AL | 25/90 | |
| PAL14H4A | 25/90 | | | |
| PAL16C1A | 25/90 | | | Features: |
| PAL10L8A | 25/90 | | | • tPD = 25 ns |
| PAL12L6A | 25/90 | | | • Zero standby power = 100 µA |
| PAL14L4A | 25/90 | | | • Operating supply current = 5 mA + 3 mA/MHz |
| PAL16L2A | 25/90 | | | • * f = 1 MHz |
| PAL10H8A2 | 35/45 | PAL10H8 | 35/90 | Can be replaced by ZHAL20A series. |
| | | PAL10H8-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | |
| PAL12H6A2 | 35/45 | PAL12H6 | 35/90 | Features: |
| | | PAL12H6-2 | 60/45 | • tPD = 25 ns |
| | | AmPAL18P8Q | 35/55 | • Standby supply current = 100 µA |
| PAL14H4A2 | 35/45 | PAL14H4 | 35/90 | • Operating supply current = 5 mA + 3 mA/MHz |
| | | PAL14H4-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | Can be replaced by AmPAL18P8Q. |
| PAL16C1A2 | 40/45 | PAL16C1 | 40/90 | |
| | | PAL16C1-2 | 60/45 | |
| PAL10L8A2 | 35/45 | PAL10L8 | 35/90 | |
| | | PAL10L8-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | |
| PAL12L6A2 | 35/45 | PAL12L6 | 35/90 | |
| | | PAL12L6-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | |
| PAL14L4A2 | 35/45 | PAL14L4 | 35/90 | |
| | | PAL14L4-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | |
| PAL16L2A2 | 35/45 | PAL16L2 | 35/90 | |
| | | PAL16L2-2 | 60/45 | |
| | | AmPAL18P8Q | 35/55 | |
| PAL16L8 | 35/180 | AmPAL16L8 | 35/155 | |
| PAL16R8 | 35/180 | AmPAL16R8 | 35/180 | |
| PAL16R6 | 35/180 | AmPAL16R6 | 35/180 | |
| PAL16R4 | 35/180 | AmPAL16R4 | 35/180 | |
| PAL16L8A | 25/180 | Am/PAL16L8A | 25/180 | AmPAL16L8A at 155 mA |
| PAL16R8A | 25/180 | Am/PAL16R8A | 25/180 | |
| PAL16R6A | 25/180 | Am/PAL16R6A | 25/180 | |
| PAL16R4A | 25/180 | Am/PAL16R4A | 25/180 | |

## Natonal Semiconductor Corporation (Cont'd.)

| NSC | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| PAL16L8A2<br>PAL16R8A2<br>PAL16R6A2<br>PAL16R4A2 | 35/90<br>35/90<br>35/90<br>35/90 | Am/PAL16L8A-2/L<br>Am/PAL16R8A-2/L<br>Am/PAL16R6A-2/L<br>Am/PAL16R4A-2/L | 35/90<br>35/90<br>35/90<br>35/90 | AmPAL16L8L at 80 mA |
| PAL16L8B<br>PAL16R8B<br>PAL16R6B<br>PAL16R4B | 15/180<br>15/180<br>15/180<br>15/180 | Am/PAL16L8B<br>Am/PAL16R8B<br>Am/PAL16R6B<br>Am/PAL16R4B | 15/180<br>15/180<br>15/180<br>15/180 | |
| PAL16L8B2<br>PAL16R8B2<br>PAL16R6B2<br>PAL16R4B2 | 25/90<br>25/90<br>25/90<br>25/90 | Am/PAL16L8B-2/AL<br>Am/PAL16R8B-2/AL<br>Am/PAL16R6B-2/AL<br>Am/PAL16R4B-2/AL | 25/90<br>25/90<br>25/90<br>25/90 | |
| PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 25/180<br>25/180<br>25/180<br>25/180 | PAL16P8A<br>PAL16RP8A<br>PAL16RP6A<br>PAL16RP4A | 25/30*/180<br>25/30*/180<br>25/30*/180<br>25/30*/180 | *Polarity fuse programmed high<br>*Polarity fuse programmed high<br>*Polarity fuse programmed high<br>*Polarity fuse programmed high |
| PAL12L10<br>PAL14L8<br>PAL16L6<br>PAL18L4<br>PAL20L2<br>PAL20C1 | 40/100<br>40/100<br>40/100<br>40/100<br>40/100<br>40/100 | PAL12L10<br>PAL14L8<br>PAL16L6<br>PAL18L4<br>PAL20L2<br>PAL20C1 | 40/100<br>40/100<br>40/100<br>40/100<br>40/100<br>40/100 | All except 20C1 can be replaced by AmPAL22P10 |
| PAL20L8A<br>PAL20R8A<br>PAL20R6A<br>PAL20R4A | 25/210<br>25/210<br>25/210<br>25/210 | PAL20L8A<br>PAL20R8A<br>PAL20R6A<br>PAL20R4A | 25/210<br>25/210<br>25/210<br>25/210 | |
| PAL20L10<br>PAL20X10<br>PAL20X8<br>PAL20X4 | 50/165<br>50/180<br>50/180<br>50/180 | PAL20L10A<br>PAL20X10A<br>PAL20X8A<br>PAL20X4A | 30/165<br>30/180<br>30/180<br>30/180 | |
| PAL1016P8 | 6/240 | PAL10H20P8 | 6/210 | NSC has 8 PT's/output. MMI has 4 (up to 8 PT's) plus 4 extra array inputs. MMI is much less expensive. |
| PAL10016P8 | 6/240 | PAL10020GR8-6 | 6/220 | 100K ECL PAL IC; available 1Q88 |

## Signetics

| Signetics | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| PLS153<br>PLS153A | 40/155<br>30/155 | PAL10H8<br>PAL10L8<br>PAL12H6<br>PAL12L6<br>PAL14H4<br>PAL14L4<br>PAL16H2<br>PAL16L2<br>PAL16C1<br>PAL16P8A<br>AmPAL18P8AL | 35/90<br>35/90<br>35/90<br>35/90<br>35/90<br>35/90<br>35/90<br>35/90<br>35/90<br>25/30*/180<br>25/90 | Signetics' parts are functional three-state output supersets of the 20 pin devices listed. Signetics' parts have 42 PT's vs. 16 PT's (64 for the 16P8A/18P8)<br>Signetics' part has PLA architecture, AMD/MMI part has PAL device architecture.<br><br>*Polarity fused programmed high. |
| PLS155 | 50/190 | AmPAL23S8-20<br>AmPAL16R4<br>Am/PAL16R4A<br>PAL16RP4A<br>PALC16R4Q-25 | 20/200<br>35/180<br>25/180<br>25/180<br>25/45 | The registered Signetics' parts are not direct functional supersets of PAL devices listed. There are important architectural differences. The Signetics' parts may not be able to implement the equivalent logic that can be implemented with a PAL device. |
| PLS157 | 50/190 | AmPAL23S8-20<br>AmPAL16R6<br>Am/PAL16R6A<br>PAL16RP6A<br>PALC16R6Q-25 | 20/200<br>35/180<br>25/180<br>25/180<br>25/45 | |
| PLS159<br>PLS159A | 50/190<br>35/190 | AmPAL23S8-20<br>Am/PAL16R8<br>Am/PAL16R8A<br>PAL16RP8A<br>PALC16R8Q-25 | 20/200<br>35/180<br>25/180<br>25/180<br>25/45 | |
| PLS173 | 30/170 | AmPAL22P10AL<br>PAL12L10<br>PAL14L8<br>PAL16L6<br>PAL18L4<br>PAL20L2<br>PAL20C1<br>PAL20L10A | 25/90<br>40/100<br>40/100<br>40/100<br>40/100<br>40/100<br>40/100<br>30/165 | |
| PLS179 | 35/210 | PAL20R8A<br>PAL20RS10<br>PALC20R8Z-35 | 25/210<br>35/40*/240<br>35/10** | MMI devices have 64 PTs vs. 42. MMI's are fixed, except on 20RS10 family, where they are "steerable".<br>Signetics' part has PLA architecture, AMD/MMI part has PAL device architecture.<br>* Polarity fuse programmed high<br>** f = 1MHz |
| PLHS18P8A | 20/155 | AmPAL18P8B<br>AmPAL18P8AL | 15/180<br>25/90 | |
| PLS105A | 20MHz/180 | PLS105-37 | 37MHz/180 | Has 16 inputs, 48 states, 8 registers. 28 pin pkg. 600 mil wide |
| PLS167A | 20MHz/180 | PLS167-33 | 33MHz/180 | Has 14 inputs, 48 states, 6 registers. 24 pin pkg. 300 mil wide |
| PLS168A | 20MHz/180 | PLS168-33 | 33MHz/180 | Has 12 inputs, 48 states, 8 registers. 24 pin pkg. 300 mil wide<br><br>Note that MMI programmable sequencers are 13 MHz faster than Signetics' and will be available 4Q87. |

## Texas Instruments

| Texas Instruments | | AMD/MMI | | COMMENTS |
|---|---|---|---|---|
| Part Number | Speed/Power | Part Number | Speed/Power | |
| TIBPAL16L8-10 | 10/180 | Am/PAL16L8D | 10/180 | |
| TIBPAL16R8-10 | 10/180 | Am/PAL16R8D | 10/180 | |
| TIBPAL16R6-10 | 10/180 | Am/PAL16R6D | 10/180 | |
| TIBPAL16R4-10 | 10/180 | Am/PAL16R4D | 10/180 | |
| TIBPAL16L8-15 | 15/180 | Am/PAL16L8B | 15/180 | |
| TIBPAL16R8-15 | 15/180 | Am/PAL16R8B | 15/180 | |
| TIBPAL16R6-15 | 15/180 | Am/PAL16R6B | 15/180 | |
| TIBPAL16R4-15 | 15/180 | Am/PAL16R4B | 15/180 | |
| TIBPAL16L8-25 | 25/100 | AmPAL16L8AL | 25/90 | |
| TIBPAL16R8-25 | 25/100 | AmPAL16R8AL | 25/90 | |
| TIBPAL16R6-25 | 25/100 | AmPAL16R6AL | 25/90 | |
| TIBPAL16R4-25 | 25/100 | AmPAL16R4AL | 25/90 | |
| PAL16L8A | 25/180 | Am/PAL16L8A | 25/180 | AmPAL16L8A at 155 mA |
| PAL16R8A | 25/180 | Am/PAL16R8A | 25/180 | |
| PAL16R6A | 25/180 | Am/PAL16R6A | 25/180 | |
| PAL16R4A | 25/180 | Am/PAL16R4A | 25/180 | |
| PAL16L8A-2 | 35/90 | Am/PAL16L8A-2/L | 35/90 | AmPAL16L8L at 80 mA |
| PAL16R8A-2 | 35/90 | Am/PAL16R8A-2/L | 35/90 | |
| PAL16R6A-2 | 35/90 | Am/PAL16R6A-2/L | 35/90 | |
| PAL16R4A-2 | 35/90 | AmPAL16R4A-2/L | 35/90 | |
| PAL16L8 | 35/180 | AmPAL16L8 | 35/155 | |
| PAL16R8 | 35/180 | AmPAL16R8 | 35/180 | |
| PAL16R6 | 35/180 | AmPAL16R6 | 35/180 | |
| PAL16R4 | 35/180 | AmPAL16R4 | 35/180 | |
| TIBPAL20L8-15 | 15/180 | PAL20L8B | 15/210 | |
| TIBPAL20R8-15 | 15/180 | PAL20R8B | 15/210 | |
| TIBPAL20R6-15 | 15/180 | PAL20R6B | 15/210 | |
| TIBPAL20R4-15 | 15/180 | PAL20R4B | 15/210 | |
| TIBPAL20L10-20 | 20/165 | AmPAL20L10-20 | 20/165 | Can be replaced by ZHAL24A Series.<br>Features:<br>• tPD = 25 ns<br>• Standby supply current = 100 µA<br>• Operating supply current = 5 mA + 3 mA/MHz |
| TIBPAL20X10-20 | 20/180 | AmPAL20XRP10-20 | 20/210 | |
| TIBPAL20X8-20 | 20/180 | AmPAL20XRP8-20 | 20/210 | |
| TIBPAL20X4-20 | 20/180 | AmPAL20XRP4-20 | 20/210 | |
| PAL22V10A | 20/180 | PALC22V10H-25 | 25/90 | |
| | | AmPAL22V10-15 | 15/180 | |
| TIBPALR19L8-25 | 25/210 | AmPALC29M16-35 | 35/120 | Input register device |
| TIBPALR19R8-25 | 25/210 | | | |
| TIBPALR19R6-25 | 25/210 | | | |
| TIBPALR19R4-25 | 25/210 | | | |
| TIBPALT19L8-25 | 25/210 | AmPALC29M16-35 | 35/120 | Input latch device |
| TIBPALT19R8-25 | 25/210 | | | |
| TIBPALT19R6-25 | 25/210 | | | |
| TIBPALT19R4-25 | 25/210 | | | |
| TIBPAL16L8-12 | 12/200 | Am/PAL16L8D | 10/180 | |
| TIBPAL16R8-12 | 12/200 | Am/PAL16R8D | 10/180 | |
| TIBPAL16R6-12 | 12/200 | Am/PAL16R6D | 10/180 | |
| TIBPAL16R4-12 | 12/200 | Am/PAL16R4D | 10/180 | |

# Worldwide Application Support

The Worldwide Application Support Group is dedicated to making sure that the customer's need for technical support is met.

It is the job of Applications Engineering to ensure that the customer engineer is familiar with our product line, that they have the information and tools necessary to get the appropriate part designed in, and that they have access to technical support throughout the lifetime of their product.

Support is provided by two groups:

**Field Applications Engineers**—With Applications Engineers located in our field sales offices throughout the world, our customers always have access to an engineer when they have problems or questions. Working with Sales, our engineers teach seminars showing customers how to use our new parts, assist them during their design when technical questions come up, and provide troubleshooting support to help eliminate problems which might occur when the design reaches production. Call your local Sales Office (page 6-41) for an FAE in your area.

**Factory Applications Support**—This group, located within the factory, is responsible for centralized technical support of all AMD's products. Their duties include developing seminars for training of FAEs and customers, and hosting the twice yearly Applications Conference. They are responsible for the coordination of customer training and giving the factory an insight into the customer's point of view.

Additionally, the Factory Applications Support Group provides customer support for technical questions via a toll-free number: (800) 222-9323 (listed on the back of all databooks). This department is manned from 7:00 A.M. to 5:30 P.M. (Pacific time) Monday to Friday. Currently 80% of the questions are answered on the first call, 20% within 24 hours.

## ADVANCED MICRO DEVICES' NORTH AMERICAN SALES OFFICES

ALABAMA ..... (205) 882-9122
ARIZONA ..... (602) 242-4400
CALIFORNIA,
Culver City ..... (213) 645-1524
Newport Beach ..... (714) 752-6262
San Diego ..... (619) 560-7030
San Jose ..... (408) 249-7766
Santa Clara ..... (408) 727-3270
Woodland Hills ..... (818) 992-4155
CANADA, Ontario,
Kanata ..... (613) 592-0060
Willowdale ..... (416) 224-5193
COLORADO ..... (303) 741-2900
CONNECTICUT ..... (203) 264-7800
FLORIDA,
Clearwater ..... (813) 530-9971
Ft Lauderdale ..... (305) 776-2001
Melbourne ..... (305) 729-0496
Orlando ..... (305) 859-0831
GEORGIA ..... (404) 449-7920
ILLINOIS,
Chicago ..... (312) 773-4422
Naperville ..... (312) 505-9517
INDIANA ..... (317) 244-7207
KANSAS ..... (913) 451-3115
MARYLAND ..... (301) 796-9310
MASSACHUSETTS ..... (617) 273-3970
MINNESOTA ..... (612) 938-0001
MISSOURI ..... (314) 275-4415
NEW JERSEY ..... (201) 299-0002
NEW YORK,
Liverpool ..... (315) 457-5400
Poughkeepsie ..... (914) 471-8180
Woodbury ..... (516) 364-8020
NORTH CAROLINA ..... (919) 847-8471
OHIO ..... (614) 891-6455
Columbus ..... (614) 891-6455
Dayton ..... (513) 439-0470
OREGON ..... (503) 245-0080
PENNSYLVANIA,
Allentown ..... (215) 398-8006
Willow Grove ..... (215) 657-3101
TEXAS,
Austin ..... (512) 346-7830
Dallas ..... (214) 934-9099
Houston ..... (713) 785-9001
WASHINGTON ..... (206) 455-3600
WISCONSIN ..... (414) 792-0590

## ADVANCED MICRO DEVICES' INTERNATIONAL SALES OFFICES

BELGIUM,
Bruxelles ..... TEL ..... (02) 771 91 42
FAX ..... (02) 762 37 12
TLX ..... 61028
FRANCE,
Paris ..... TEL ..... (1) 49-75-10-10
FAX ..... (1) 49-75-10-13
TLX ..... 263282
GERMANY,
Hannover area ..... TEL ..... (05143) 50 55
FAX ..... (05143) 55 53
TLX ..... 925287
München ..... TEL ..... (089) 41 14-0
FAX ..... (089) 406490
TLX ..... 523883
Stuttgart ..... TEL ..... (0711) 62 33 77
FAX ..... (0711) 625187
TLX ..... 721882
HONG KONG,
Kowloon ..... TEL ..... 852-3-695377
FAX ..... 852-123-4276
TLX .. 504260AMDAPHX
ITALY, Milano ..... TEL ..... (02) 3390541
(02) 3533241
FAX ..... (02) 3498000
TLX ..... 315286
JAPAN,
Tokyo ..... TEL ..... (03) 345-8241
FAX ..... (03) 342-5196
TLX ... J24064AMDTKOJ
Osaka ..... TEL ..... 06-243-3250
FAX ..... 06-243-3253
KOREA, Seoul ..... TEL ..... 82-2-784-7598
FAX ..... 82-2-784-8014
LATIN AMERICA,
Ft. Lauderdale ..... TEL ..... (305) 484-8600
FAX ..... (305) 485-9736
TLX .. 5109554261 AMDFTL
NORWAY,
Hovik ..... TEL ..... (02) 537810
FAX ..... (02) 591959
TLX ..... 79079
SINGAPORE ..... TEL ..... 65-2257544
FAX ..... 2246113
TLX ..... RS55650 MMI RS
SWEDEN, Stockholm ..... TEL ..... (08) 733 03 50
FAX ..... (08) 733 22 85
TLX ..... 11602
TAIWAN ..... TLX ..... 886-2-7122066
FAX ..... 886-2-7122017
UNITED KINGDOM,
Farnborough ..... TEL ..... (0252) 517431
FAX ..... (0252) 521041
TLX ..... 858051
Manchester area ..... TEL ..... (0925) 828008
FAX ..... (0925) 827693
TLX ..... 628524
London area ..... TEL ..... (04862) 22121
FAX ..... (04862) 22179
TLX ..... 859103

## NORTH AMERICAN REPRESENTATIVES

CALIFORNIA
$I^2$ INC ..... OEM (408) 988-3400
DISTI (408) 498-6868
CANADA
Calgary, Alberta
VITEL ELECTRONICS ..... (403) 278-5833
Kanata, Ontario
VITEL ELECTRONICS ..... (613) 592-0090
Mississauga, Ontario
VITAL ELECTRONICS ..... (416) 676-9720
Quebec
VITEL ELECTRONICS ..... (514) 636-5951
IDAHO
INTERMOUNTAIN TECH MKGT ..... (208) 888-6071
INDIANA
SAI MARKETING CORP ..... (317) 253-1668
IOWA
LORENZ SALES ..... (319) 377-4666
KANSAS
LORENZ SALES ..... (913) 384-6556
MICHIGAN
SAI MARKETING CORP ..... (313) 750-1922
MISSOURI
LORENZ SALES ..... (314) 997-4558
NEBRASKA
LORENZ SALES ..... (402) 475-4660
NEW MEXICO
THORSON DESERT STATES ..... (505) 293-8555
NEW YORK
NYCOM, INC ..... (315) 437-8343
OHIO
Columbus
DOLFUSS ROOT & CO ..... (614) 885-4844
Dayton
DOLFUSS ROOT & CO ..... (513) 433-6776
Strongsville
DOLFUSS ROOT & CO ..... (216) 238-0300
PENNSYLVANIA
DOLFUSS ROOT & CO ..... (412) 221-4420
UTAH
$R^2$ MARKETING ..... (801) 595-0631

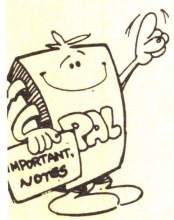
Advanced Micro Devices reserves the right to make changes in its product without notice in order to improve design or performance characteristics. The performance characteristics listed in this document are guaranteed by specific tests, guard banding, design and other practices common to the industry. For specific testing details, contact your local AMD sales representative. The company assumes no responsibility for the use of any circuits described herein.

***ADVANCED MICRO DEVICES*** 901 Thompson Pl., P.O. Box 3453, Sunnyvale, CA 94088, USA
TEL: (408) 732-2400 • TWX: 910-339-9280 • TELEX: 34-6306 • TOLL FREE: (800) 538-8450
***APPLICATIONS HOTLINE*** TOLL FREE: (800) 222-9323

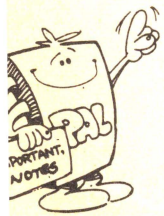
© 1988 Advanced Micro Devices, Inc.

B-90M-2/88-0

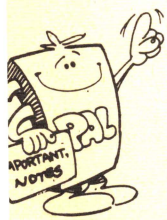

IMPORTANT
NOTES


IMPORTANT
NOTES

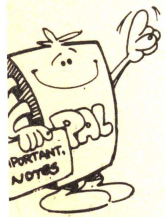

IMPORTANT
NOTES


PORTANT
NOTES


IMPORTANT
NOTES


PORTANT
NOTES


IMPORTANT
NOTES


IPORTANT.
NOTES


IMPORTANT
NOTES

## 20-pin Combinatorial TTL/CMOS PAL Devices

| PRODUCT | PINS | TECHNOLOGY | $t_{PD}$(ns) | $I_{CC}$(mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| AmPAL18P8 | 20 | TTL | 15, 25 | 55, 90, 180 | 20-pin superset | 5-202 |
| PAL16P8 | 20 | TTL | 25 | 180 | Programmable polarity | 5-17 |
| Am/PAL16L8 | 20 | TTL, E CMOS | 10, 15, 25, 35, 55 | 0.1, 45, 90, 180 | Standard | 5-26, 5-184 |
| PAL10H8 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL10L8 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL12H6 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL12L6 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL14H4 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL14L4 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL16H2 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL16L2 | 20 | TTL | 35 | 90 | Simple combinatorial | 5-56 |
| PAL16C1 | 20 | TTL | 40 | 0 | Simple combinatorial | 5-56 |

## 20-pin Registered TTL/CMOS PAL Devices

| PRODUCT | PINS | TECHNOLOGY | $f_{MAX}$(MHz) | $I_{CC}$(mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| AmPAL23S8 | 20 | TTL | 33, 28.5 | 200 | Buried registers | 5-169 |
| PAL16RA8 | 20 | TTL | 20 | 170 | Asynchronous | 5-11 |
| PAL16RP8 | 20 | TTL | 25 | 180 | Programmable polarity | 5-17 |
| PAL16RP6 | 20 | TTL | 25 | 180 | Programmable polarity | 5-17 |
| PAL16RP4 | 20 | TTL | 25 | 180 | Programmable polarity | 5-17 |
| Am/PAL16R8 | 20 | TTL, E CMOS | 58, 40, 28.5, 18, 11 | 0.1, 45, 90, 180 | Standard | 5-26, 5-184 |
| Am/PAL16R6 | 20 | TTL, E CMOS | 58, 40, 28.5, 18, 11 | 0.1, 45, 90, 180 | Standard | 5-26, 5-184 |
| Am/PAL16R4 | 20 | TTL, E CMOS | 58, 40, 28.5, 18, 11 | 0.1, 45, 90, 180 | Standard | 5-26, 5-184 |
| PAL16X4 | 20 | TTL | 14 | 225 | Arithmetic | 5-51 |

## Sequencers

| PRODUCT | PINS | TECHNOLOGY | $f_{MAX}$(MHz) | $I_{CC}$(mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| PMS14R21 | 24 | TTL | 30, 25 | 210 | PROSE Sequencer | 5-315 |
| PLS167 | 24 | TTL | 33 | 200 | Programmable Logic Seq. | 5-331 |
| PLS168 | 24 | TTL | 33 | 200 | Programmable Logic Seq. | 5-331 |
| PLS105 | 24 | TTL | 37 | 200 | Programmable Logic Seq. | 5-331 |
| Am29PL141 | 28 | TTL | 20 | 450 | Fuse Prog. Controller | 5-339 |
| Am2971 | 24 | TTL | 85 | 310 | Prog. Event Generator | 5-365 |

## ECL PAL Devices

| PRODUCT | PINS | TECHNOLOGY | $f_{MAX}/t_{PD}$ | $I_{CC}$(mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| PAL10H20EV/EG8 | 24 | ECL 10KH | 125 MHz | 220 | Registered/latched | 5-381 |
| PAL10020EV/EG8 | 24 | ECL 100K | 125 MHz | 220 | Registered/latched | 5-381 |
| PAL10H20G8 | 24 | ECL 10KH | 6 ns | 225 | Latched | 5-382 |
| PAL10H20P8 | 24 | ECL 10KH | 6 ns | 210 | Combinatorial | 5-386 |

## Logic Cell Array

| PRODUCT | PINS | TECHNOLOGY | $f_{MAX}$(MHz) | $I_{CC}$(mA) | DESCRIPTION | PAGE |
|---|---|---|---|---|---|---|
| M2064 | 48,68 | RAM CMOS | 70, 50, 33 | 5 | Logic Cell Array | 5-483 |
| M2018 | 68,84 | RAM CMOS | 50, 33 | 5 | Logic Cell Array | 5-483 |

(Continued on front cover)



**Advanced**
**Micro**
**Devices**
901 Thompson Place
P.O. Box 3453
Sunnyvale
California 94088-3453
(408) 732-2400
TELEX: 34-6306
**TOLL FREE (800) 538-8450**
**APPLICATIONS HOTLINE**
**(800) 222-9323**

© 1988 Advanced Micro
Devices, Inc.
Printed in USA

10173A/0